Business Suitability Principles for Workflow Modelling*

A.P. Barros[1], A.H.M. ter Hofstede[1], H.A. Proper[2], P.N. Creasy[1]

[1]Department of Computer Science
The University of Queensland
Brisbane, QLD 4072
Australia

[2]Faculty of Information Technology
Queensland University of Technology
GPO Box 2434, Brisbane QLD 4001
Australia
e-mail: E.Proper@acm.org

Keywords: Conceptual Modelling, Workflow, Business Modelling, Object-Oriented Analysis

PUBLISHED AS:

A.P. Barros, A.H.M. ter Hofstede, H.A. Proper, and P.N. Creasy. Business Suitability Principles for Workflow Modelling. Technical Report 380, Department of Computer Science, University of Queensland, Brisbane, Australia, August 1996.

Abstract

By incorporating aspects of coordination and collaboration, workflow implementations of information systems require a sound conceptualisation of *business processing* semantics. Traditionally, the success of conceptual modelling techniques has depended largely on the adequacy of conceptualisation, expressive power, comprehensibility and formal foundation. An equally important requirement, particularly with the increased conceptualisation of business aspects, is *business suitability*.

In this paper, the focus is on the business suitability of workflow modelling for a commonly encountered class of (operational) business processing, e.g. those of insurance claims, bank loans and land conveyancing. A general assessment is first conducted on some *integrated* techniques characterising well-known paradigms - structured process modelling, object-oriented modelling, behavioural process modelling and business-oriented modelling. Through this, an insight into business suitability within the broader perspective of technique adequacy, is gained. A specific business suitability diagnosis then follows using a particular characterisation of business processing, i.e. one where the intuitive semantics and inter-relationship of business services and business processes are nuanced. As a result, five business suitability principles are elicited. These are proposed for a more detailed understanding and (synthetic) development of workflow modelling techniques. Accordingly, further insight into workflow specification languages and workflow globalisation in open distributed architectures may also be gained.

*Part of this work has been supported by CITEC, a business unit of the Queensland Government's Department of Public Works and Housing (formerly the Administrative Services Department).

1 Introduction

The workflow concept, proliferated through the recently emergent computer supported cooperative work (CSCW) systems and workflow systems (see surveys in [FYW94, WW93, Rod91] and [GHS95] respectively), advances information systems (IS) implementation models by incorporating aspects of collaboration and coordination in business processes. Under traditional implementation models, applications are partitioned into discrete units of functionality, with (typically) operational procedures used to describe how human and computerised actions of business processes combine to deliver business services. Through an endowment of business process execution semantics, workflows permit a greater organisational fit of ISs. Moreover workflows are specified at a level above traditional applications, enabling program binding and access to a loosely-coupled set of databases and files. Therefore, newer applications may be developed out of existing applications to reflect reengineered business processes.

Crucial to the specification of any IS implementation is the *conceptual* level. This, of course, orients the analysis of a given domain towards its essence (deep-structure) rather than to aspects of implementation (physical-structure) or representation (surface-structure). It is a well-known fact the later problems and inadequacies are detected in specifications, the greater the expense of correction [Dav90].

For workflows, the standardisation of concepts is progressing through the Workflow Management Coalition[1]. While the set of terms and references defined so far characterise sufficiently the notion of workflow, e.g. event, process (including pre-conditions, post-conditions and state transitions) and organisational (or actor) role, much of the focus is geared towards workflow management systems and their specification languages. The emphasis is on part of business processing, namely process execution semantics: sequence, repetition, choice, parallelism and synchronisation. A sound conceptualisation requires not only this but also that process semantics, e.g. the messaging, database updates and retrievals involved, to be explicitly captured.

In general, for the conceptual level, *techniques* are available under different paradigms, for the modelling of different aspects of a business domain (see e.g. [OHM$^+$88]). When integrated into well-formed methods, integrated IS specifications - result. A number of paradigms may be discerned for workflow modelling: process-centric, e.g. [DB91]; state-centric, e.g. [DP95]; and actor-centric, e.g. [Die94] (based on the speech-act theory synthesis of [FL80]). Moreover the use of business (or enterprise) models, e.g. as deployed in requirements engineering methods [BB95, LK95, AMP94], in design methods [Ram94] and in CAiSE tools e.g. AD/CYCLE [MMNR90], provides an *organisational embedding* whereby a workflow model's components may be backtracked to its real-world counterparts.

Although, the field of conceptual modelling has become fairly mature, the application of techniques, has, by and large, followed the intuition of the developers of models. This, of course, involves an informal to formal transition. With workflow specifications, this transition is reduced, however a greater alignment is required between the workflow modelling cognition and business processing cognition. Beyond the qualification of fundamental modelling concepts (e.g. process) with organisational attributes (e.g. business service), the business processing semantics need to be infused into the semantics of a technique such that a workflow may be expressed and communicated adequately using that technique. In absence of a universal organisational theory, much uncertainty exists as to how effective conceptual modelling techniques are for business workflows; whether, given the diversity of business processing, any generable prescription of business processing cognition is in fact possible or desirable.

[1] Refer to http://www.aiai.ed.ac.uk/WfMC/index.html for more details.

This paper addresses some of this uncertainty. It recognises that there are a basic set of requirements which conceptual modelling techniques should fulfill in order to be effective. These requirements are based on the well-known conceptual modelling principles of [ISO87]. If not properly catered for, the development of workflow specifications may be problematic.

First and foremost, in accordance with the *Conceptualisation Principle*, a technique should focus on essential detail only. The focal aspects of business models and workflow specification languages can deflect and sometimes dictate the quality of conceptualisation. Conversely, following the *One Hundred Percent Principle*, techniques should provide a sufficient *expressive power* so that a full conceptualisation is in fact possible. If the expressiveness of a workflow specification language exceeds that of a technique, where only a partial conceptualisation results, the remainder has to be addressed at the implementation level. This problem is known in software engineering jargon as the *waterfall*. Such a disjointness of analysis precludes a sound understanding of the problem, leading to premature implementations. Conceptual models also need to be communicated and validated with a diverse set of stakeholders and so mechanisms are required for an effective *comprehensibility*. An effective graphical presentation together with abstraction/decomposition mechanisms facilitates this. At the same time, a *formal foundation* is required to prevent interpretation ambiguities and to enable formal reasoning. That a recent survey on workflow technology [GHS95] cited the problem that "workflow models and process methodologies do not explicitly support the specification of what it means for a workflow to be correct" is symptomatic of a lack of formal semantics. Together with a formal syntax, this constitutes formal foundation. Finally and most relatedly to the issue of the paper, since a "silver bullet" for all types of domains is considered unrealistic (see e.g. [BS87, Bro87, ML83]), techniques should be *suitable* for their problem domains. This means a close connection between the modelling concepts and features, and those required by the domain.

Of significance for business processing domains is the *business suitability* of techniques. Since there are many types of organisations and many types of business processing [DO85], particular attention is drawn to that type of (operational) business processing which exhibits precise execution paths. As examples, the processing of insurance claims, bank loans and land conveyancing, are mission-critical in nature and are rarely undertaken without strict operational procedure. Also, multiple interactions with clients and external organisations are typically needed to fulfill service requests. Given their closer connection with database technology, workflow systems are considered more appropriate for this type of business processing over CSCW [McC92, GHS95]. These notions of business processing, workflow and workflow systems are assumed hereafter.

Following the constructivist approach, notably as advocated for IS concept development in [FHL$^+$98], a number of *integrated* conceptual modelling techniques - are assessed using the aforementioned requirements. Attention then turns to determining common problems of business suitability, and in turn, the formulation of a number of principles. The result is five principles. The *Organisational Embedding Principle* describes how a model should be backtracked to organisational elements. The *Scenario Validation Principle* identifies the need for scenarios, and in particular, a business transaction, as distinct from business process and business service, for workflow cognition. The *Service Information Hiding Principle* requires that business processing undertaken for business service requests should be insulated from the requests, and in so doing, motivates the need for an explicit treatment of business services within conceptual modelling. The *Cognitive Sufficiency Principle* requires that all concepts involved in workflow model enactment be simultaneously present in the model. Simultaneously absent in the assessed techniques were the combination of structural *and* behavioural aspects of workflows, human to computer interaction and temporal aspects. Finally, the *Execution Resilience Principle* identifies

the need for operational error handling to be catered for at the conceptual level, thereby incorporating the recovery management focus of *transactional* workflows into a general exception handling. An improved insight into the assessment and development of workflow modelling techniques, is claimed. Also the proposed separation of business service, business process and business transaction, can lead to a more effective globalisation of workflows, where, for example, within open distributed environments (e.g. the Web), services involving workflows can be "mixed and matched".

The paper is organised as follows. In section 2, techniques characteristic of the different paradigms - structured process modelling, object-oriented modelling, behavioural process modelling and business-oriented modelling - are assessed. In section 3, the problems and business suitability principles are defined. In section 4, the paper is concludes with an epilogue.

2 An assessment of integrated conceptual modelling techniques

In this section, an insight into the capabilities of techniques to support a sound conceptualisation of business processing for workflow specifications is sought. For this, an assessment of some integrated conceptual modelling techniques is conducted using the general conceptual modelling requirements (described in section 1). These are: conceptualisation, expressive power, formal foundation, comprehensibility and business suitability. The techniques are drawn from the well recognised paradigms of:

- structured process modelling (section 2.1)
- object-oriented modelling (section 2.2)
- behavioural process modelling (section 2.3)
- business-oriented modelling (section 2.4)

The approach to the assessment is qualitative, reflecting its motivation. For this, the key areas of observation include integration strategy, integration structure and the mechanisms used to adapt concepts and features for the business level. An integration strategy reflects the cognitive dependency of a technique's partial models for an overall understanding of a business domain. Process modelling techniques centralise processes with respect to data flow, control flow and data repositories, and so a *process-centric* integration with data models follows. The dual of this is a *state-centric* integration, where object modelling techniques centralise object states with respect to processes involved in state transitions, and so process modelling follows.

An integration structure relates to the definition of concepts in the partial models. In a *tightly-coupled* structure, all concepts are bound to a single logical definition. In a *loosely-coupled* structure, each partial model has a separate definition, and some concepts may be integrated for basic consistency. Clearly, a tight-coupling permits a stronger expressive power and formal foundation. However, a loose-coupling provides greater flexibility - particularly for "contingency frameworks" [AW91] - where different techniques may be selected depending on the analysis "situation". Since a (relatively) specific business processing situation underscores this paper, the former is preferred in the assessment.

2.1 Structured Process Modelling

Structured process modelling, for example Structured Analysis [GS86, You89, DeM78] and ISAC [LGN81] have had widespread use, providing both top-down process analysis and software design mapping. With the prevalence of database technology, a number of extensions have been proposed, e.g. [Sho91, You89], to incorporate data modelling techniques. In particular, process models are often used for the higher levels of analysis, and the identified data repositories lead to data modelling; clearly a process-centric integration strategy.

Data Flow Diagrams (DFD), associated with Structured Analysis, are a popular - arguably the most popular - structured process modelling technique. A DFD is a directed graph where the nodes represent external entities, processes and data stores, and where the edges represent data flows between the nodes. A process may be decomposed into another DFD, provided the DFD contains at least that process's data flows. At the lowest level, each process has a detailed Structured English specification. Specifications for data flows and stores are contained in a data dictionary.

Figure 1 illustrates a process-centric integration of process and data models for a library domain using [You89]. The data model is based on an Entity Relationship Model (ERM) technique. For brevity, the data dictionary and process specifications have been omitted.

The DFD models the processing of loan requests and orders, viz: borrowers (an external entity) send loan requests (data flow) for items to LOAN (process). A loan (an update) is issued in items (data store) and the item is passed to the borrower. When the loaned item is returned to LOAN, a return is issued to the items data store to indicate that the item is available for loaning again. Borrowers also make purchase requests (for an item) to the ORDER process. For this, order details are stored in orders, and item details are stored in items. An order is sent to a publisher who supplies the ordered item. Once supplied, the item is updates to be available.

The data dictionary provides the integration between DFDs and ERMs. This is achieved using syntactic correspondences. A correspondence is required between data stores and entity types, indicated by alphabets in Figure 1. The integration structure is therefore loosely-coupled. Furthermore, it is clear that the degree of integration is coarse. For instance, no correspondence exists for relationship types, unless they are aggregated. Sometimes processes, data flows and data stores indicate relationship types as indicated by (a) and (b). However, the absence of formal semantics - an omission widely observed in most DFD techniques (see e.g. [BW89, OS93] - makes a detailed correspondence difficult. Care should also be taken when directly corresponding multi-object data stores since unnaturally aggregated entity types may result. One methodological implication is to delay ERM modelling until sufficiently detailed DFDs are drawn. Alternatively, [BCN92] proposes a joint methodology which uses mutually influential decompositions in both ERMs and DFDs.

Process specifications utilise data dictionary elements to maintain consistency with DFDs and ERMs. However, the expressive power of Structured English is restricted by the basic level of integration. Business rules including those expressed through object lifecycles are implicit in process specifications. Process dependencies are determined in an ad-hoc manner solely for the purpose of systems design.

Clearly the strength of a DFD lies in its simple and general concepts. Together with the decomposition feature, this allows an effective comprehensibility, particularly for the early phases of analysis where the broader context of a system and its scope needs to understood. Additionally, a large DFD may be partitioned by external events, permitting horizontal views for validation. Like most classical process

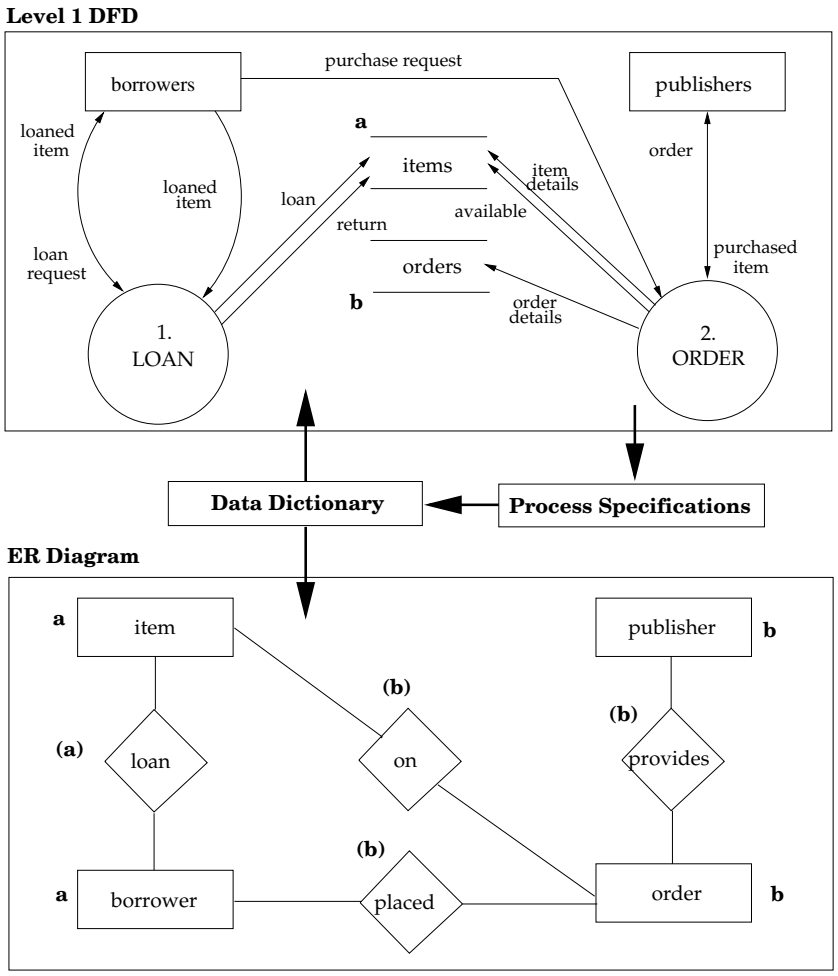

Figure 1: Process and data model integration for the library domain in Structured Analysis

modelling techniques, DFDs may be applied to several domains including real-time processing and text processing. For business domains, its methodology recommends the modelling of organisational processing structure first, from which essential DFDs are determined. In this regard, its simplicity and generality can compromise the business suitability of the technique, e.g. the relationship with organisational processing structure is arbitrary and only basic aspects of business processes are modelled. Moreover, the procedural style of expression for DFDs and process specifications imposes an imperative rather than a declarative conceptualisation. At lower levels of abstraction, DFDs can prescribe implementation, thereby compromising conceptualisation. Further problems of DFDs are cited in [OS93].

2.2 Object-oriented modelling

An object-oriented approach to programming, and in recent years, to analysis and design (OOA/D), stems from the premises that objects provide a closer semblance with reality and are less prone to change than processes. Along with static properties, objects include in their classification, dynamic properties - operations, implemented as methods on classes. In keeping with tightly-coupled object

models, conventional OOA/D techniques, [SM88, Boo91, CY90, RBP+91, JCJO92] advocate principally a state-centric integration of conceptual models. Object structure is modelled using techniques adapted, for the most part, from data modelling techniques, e.g. ERM based [EGH+92] and Object-Role Model (ORM) based [CP96]. Object behaviour is modelled using techniques based on formalisms like finite state machines (FSM) [HU79] and Petri nets [Pet81, Rei85].

Since object behaviour models describe the intra-object dynamics, a higher modelling context is required in domains where object interaction is significant. Process modelling and hence a process-centric integration can be useful for this. For example, DFDs are used in many OOA/D techniques including [Shu91, SJ91, SI90], despite the paradigmal differences which are sometimes regarded as incompatible [EJW95, Fir91]. Alternatively, other mechanisms, e.g. use cases or scenarios as in [PD94, JCJO92], have been developed to preserve a high-level state-centric context.

A popular OOA/D technique, Object Modelling Technique (OMT) [RBP+91], incorporates both a state- and process-centric integration strategy. It accommodates the data, process and behaviour perspectives through an object model (ERM), a function model (DFD) and a dynamic model (FSM which includes nesting), respectively. Figure 2 illustrates a state-centric model integration for the library domain.

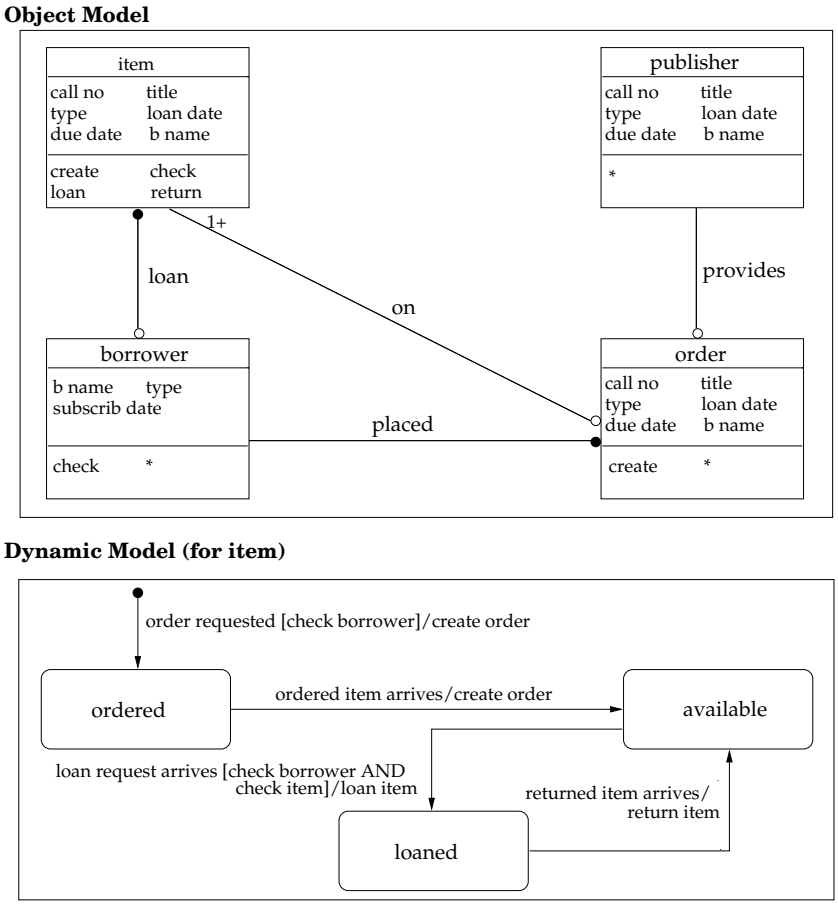

Figure 2: Object and dynamic model integration for the library domain in OMT

In the object model, attributes are listed in the middle portion of the object class diagrams and methods

are listed in the lower portion. Each object type has a dynamic model associated with it. The dynamic model contains a partial order of behaviour states as nodes. States are defined by "exclusive" predicates and are orthogonal to the object type specialisations (a structural classification). The edges represent state transitions. A state transition embodies an event which triggers an action if a condition (optional) is satisfied. The significant result of the action is, of course, the change to the next state.

In Figure 2 for example, the event order requested triggers the action create order to create an order object (having no previous state) in the state ordered if some check(s) on borrowers are satisfied[2]. A large class of business rules may be captured using state transitions. In fact, event-condition-action (ECA) paradigm for rule-based languages is adopted in active database technology, e.g. in context with object-oriented databases [CN90], and object-oriented conceptual specification languages, e.g. as defined in the ESPRIT[3] project TEMPORA [LMS$^+$91].

Digressing for a moment to another example, Figure 3, taken from [SM88] (pp. 40), illustrates a more complicated object lifecycle. An interesting feature of this example is the primary importance of state-centric integration for the domain. That is, the object behaviour model is pivotal to understanding the broader context of the domain. The pseudo code for state-transitions includes alternatives (illustrated) and iterations (not illustrated), improving the expressive power over ECA languages. In general, without a conceptual specification language, techniques either omit detailed specifications or convolute the graphical aspect of the model.

Returning to OMT, Figure 4 illustrates a process-centric integration of the function and object models for the library domain. Process-centric integration rules require that the function model's leaf processes correspond to object class methods. This is not straightforward since a single leaf process may involve more than one object type. Clearly, the principal object type referred to in OMT as the target, needs to be identified. Targets are contained in data flows, data stores and actors (represented by a DFD's external entity symbol). A target may be identified by determining a "client-server" relationship between the object types. An object type is a target if it invokes requests from other object types for some purpose related to none other of those object types. Hence the (lettered) correspondences in Figure 4; a loosely-coupled integration structure.

Like other classical OOA/D techniques, OMT provides a simple and general set of concepts for several types of domains. For business domains, comprehensibility and suitability gains of the dynamic model are useful on a per object basis. The function model addresses the "bigger picture", however process control can only be derived from the collective set of dynamic models. Despite claims of the "natural" occurrence of objects, the technique's generality can present ambiguities in the analysis of organisational processing structures. For detailed specifications, a combined cognition of the models can help resolve difficult design decisions. For example, the function model provides further insight into the aggregation of complex object types, while the dynamic and function models provide a collective insight into method identification. The lack of formal semantics (particularly for the function model) and expressive power restricts the effectiveness of these gains. In general, no evidence of formal semantics was found in the referenced OOA/D techniques.

[2]Textual aliases, in this case check borrower, have been used to denote condition predicates.

[3]ESPRIT is an acronym for European Strategic Program for Research in Information Technology.

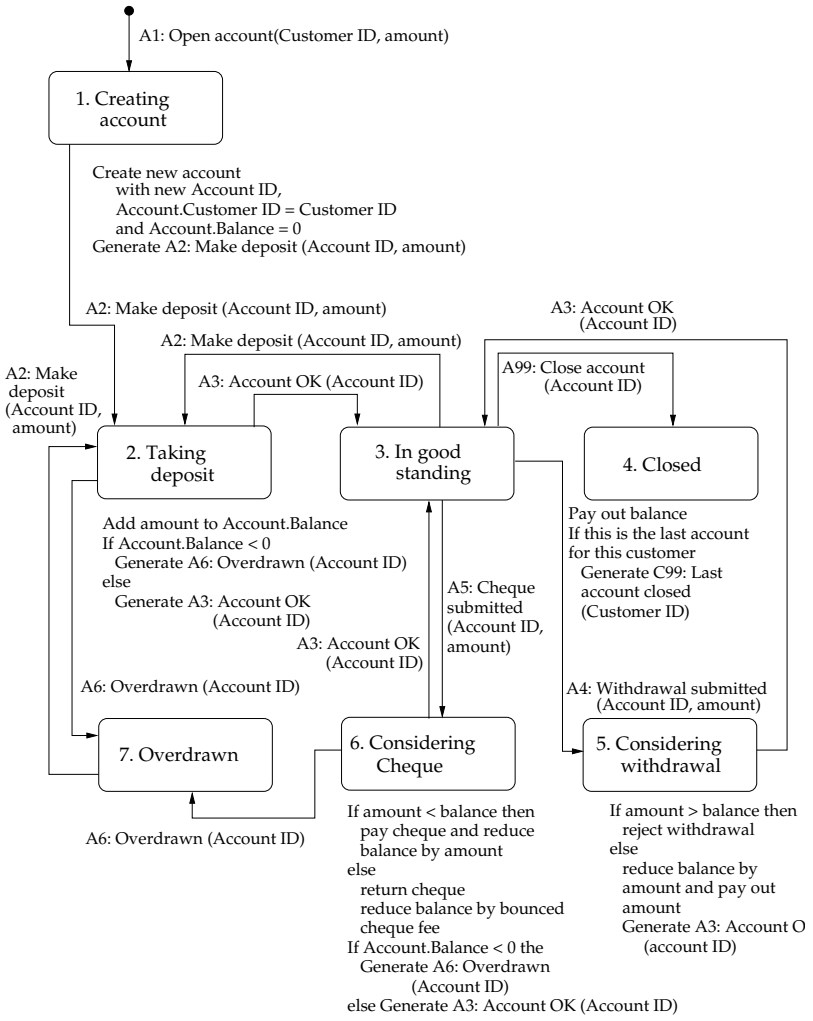

Figure 3: Object lifecycle for accounts of the bank domain in Shlaer & Mellor

2.3 Behavioural process modelling

The enhanced suitability which results from behavioural (or control) aspects of process models has already been identified in the preceding sections. Integrated techniques such as IML Inscribed Petri Nets [RD82] and Activity-Behaviour Modelling [SK86] pioneered the inclusion of behavioural process modelling in at least one level of process model decomposition. Behavioural aspects include process sequence, repetition, parallelism and synchronisation. Accordingly, collaborative aspects of processes can be modelled and problems such as deadlock, livelock and starvation may be determined prior to implementation.

Classical Petri nets, Condition/Event-nets and Place/Transition-nets, have been adapted for process behavioural specification [DZ81] but are limited in the degree of formal interpretation and real-world practicality [HL91]. High-level Petri nets such as Predicate/Transition nets (PrT-nets) [Gen87] and Coloured Petri nets (CP-nets) [Jen91] were proposed to overcome these problems by the provision of declarations and net inscriptions, expressed in a formal language (based on first-order predicate

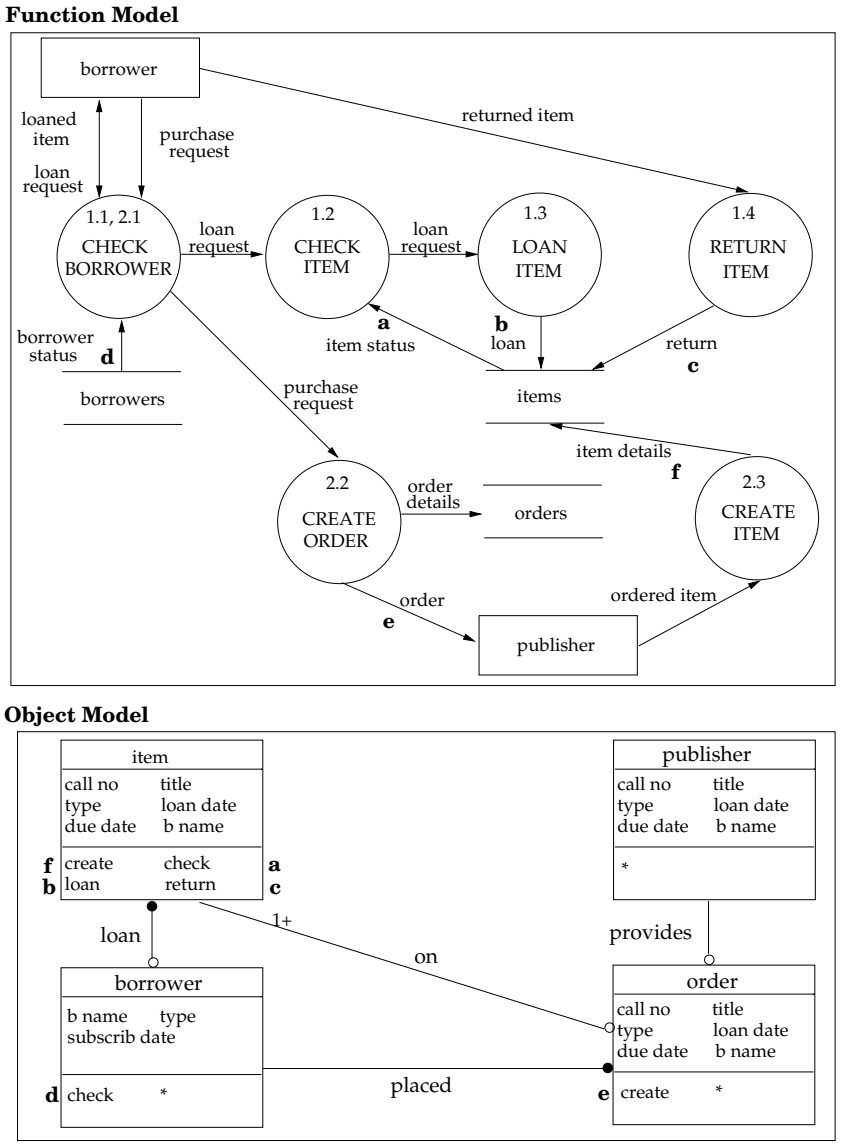

Figure 4: Function and object model integration for the library domain in OMT

logic for PrT-nets and a functional programming language for CP-nets). The incorporation of formal semantics leads to executable process specifications, e.g. in tools such as ExSpect [HSV89] (hierarchical CP-nets) and Income/Star [JOS93] (Fuzzy Nets). [Ver93] formally demonstrates an integration of an object-oriented modelling technique, Simcon, and PrT-nets. The combination of both state-centric and process-centric constructs into a single model, however, makes even high-level Petri nets difficult to comprehend. Different strategies are used to maintain comprehensibility without losing expressive power.

The Behaviour Network Model (BNM) [Kun93] transfers behavioural aspects from higher levels of abstraction into the lowest level. In it, DFDs are used at higher levels, and at the lowest level, each process is transformed into a PrT-net which is tightly-coupled with an ER schema. Thus, PrT-net specifications

replace traditional Structured English. At all levels of abstraction, model integration is process-centric. As an example, Figure 5 illustrates a DFD, PrT-net and ERM integration for the library's CHECK BORROWER process.

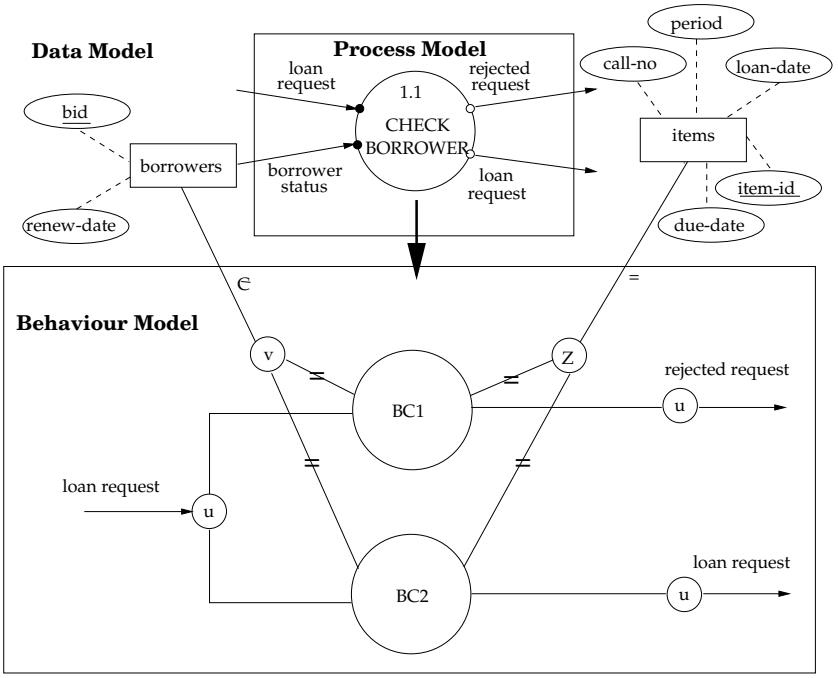

Figure 5: Data, process and behaviour model integration for the library domain in BNM

The DFD specifies that the conjunction (black dot) of loan request and borrower status are required by CHECK BORROWER so that a disjunction (white dot) of either loan request or rejected request[4] is produced. Data flows are message carriers which may be associated with ERM object types. In a PrT-net, input flows, in this case loan request, connect to places (i.e. u, v and Z) which in turn connect to transitions (i.e. BC1 and BC2). Individual (element-of sign) or sets (equality sign) of instances of ERM entity types are linked to PrT-nets via places. Transitions, representing elementary actions, use entity and data flow instances as operands in pre- and post-condition rules.

In the example, two rules are specified for CHECK BORROWER; note the declarative rule specification possible through Petri nets compared to the imperative approach of Structured English. Firstly BC1, that a loan request becomes a rejected request if either the borrower has an overdue item or will not be registered during the loan period:

$$
\begin{aligned}
pre \quad & \neg \exists_{z \in Z} \left[u.\text{bid} = z.\text{bid} \wedge z.\text{loan-date} > \$\text{TODAY} \right] \vee \\
& u.\text{bid} = v.\text{bid} \wedge v.\text{renew-date} < \text{DATE}(\$\text{TODAY}, v.\text{period}) \\
post \quad & u.\text{msg} = \text{`Loan request rejected'}
\end{aligned}
$$

and secondly BC2, to ensure that a loan request is accepted (the negation of BC1). The specification for this is not provided. Note, the variable $TODAY and the function DATE are introduced.

[4]This data flow has been introduced to allow for model execution.

Compared to Structured Analysis, BNM's DFD technique is as suitable, though improved in comprehensibility. The use of PrT-nets for the conceptualisation of process specifications improves expressive power and provides a basis for formal foundation. Process pre- and post-conditions allow a further range of business rules to be specified compared to object lifecycles. Object lifecycles however, are implicit in one or more PrT-nets, and so the related class of business rules are specified implicitly.

In Hydra [Hof93], behavioural aspects are defined in process models, task structures [HN93], at all levels of decomposition. The data modelling technique, PSM [HW93] - an ORM variant, also provides decomposition for object types (schema object types decompose into schemas). Process-centric integration is applied at at all levels of decomposition. A conceptual specification language, LISA-D [HPW93], is used to express detailed process specifications and database constraints. Figure 6 illustrates a decomposed task structure for CHECK BORROWER and the associated PSM schema.

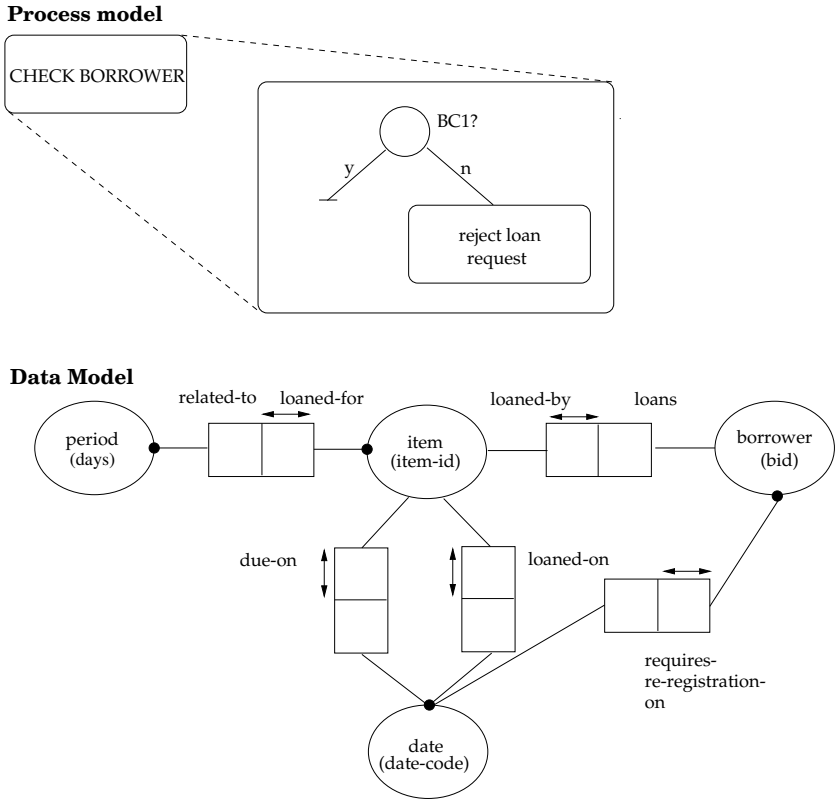

Figure 6: A task structure for CHECK BORROWER and the PSM schema

The contrast with the BNM model for the process model (Figure 5) is striking. The omission of state-based concerns simplifies considerably the process model. At the same time, sequence (arrows), decisions (circles) and synchronisations (not illustrated) provide the same expressive power as Petri net based approaches (in [HN93], a translation from Petri nets to Task Structures in provided). Of course, more complicated task structures are possible than the one bourne out by Figure 5.

Task pre- and post-conditions and decisions are specified in LISA-D. For example, the LISA-D predicate for the outgoing positive arc y) of BC1 is:

borrower B loans item due-on date $>$ \$TODAY

OR

borrower requires re-registration on date $>$ \$TODAY + period related-to item I

LISA-D expressions are formulated against the database using an attached schema. A schema is defined for the highest level task which is visible to all its decompositions. Buffers (not illustrated) and variables (B, I and \$TODAY) may also be defined for temporary storage during task processing. A schema type is attached to each buffer.

A positive outcome results in a termination of execution returning control to the calling task whereas a negative outcome results in the reject loan request being executed. Recall (from the BNM example that this should send a message through the rejected request data flow. Hydra does not have explicit data flows and so this cannot be specified. Note LISA-D's expressiveness does warrant the introduction of a DATE function since the addition operator is domain sensitive.

The chief strenght of Hydra is its expressive power and formal foundation, both exceeding that of previously assessed techniques. The formal semantics for Task Structures are described in Process Algebra [HN93]. A high comprehensibility, particularly for behavioural process modelling is also clear. A major drawback is business suitability which results from the previously discussed problem of generality. This together with the lack of data flow (including messaging) and data store constructs raises uncertainty about its effectiveness at the business level.

2.4 Business-oriented modelling

As the name suggests, business-orientation does not displace the essential nature of conceptual modelling, but rather that conceptual modelling is oriented towards an increased conceptualisation of business aspects. This shifts the general (and compromised) suitability of classical techniques, to one which is specialised for business domains. Given the lack of a standard inter-subjective world-view, the strategies for business-orientation vary significantly.

Earlier attempts at business-orientation focussed on providing some aspects of organisational processing structure, so that an *organisational embedding*[5] of conceptual models into different business world-views is possible. [RV85] was among the first to use organisational concepts such as actors and the cooperative processing of Petri nets to capture the semantics of business processes. Earlier attempts at office modelling were based on completely process-centric techniques, e.g. [Bot89, DB91]. The popularity of object concept has resulted in adaption of OOA/D techniques. In IOOM [DP95] for example, a purely state-centric approach is proposed using the office modelling concepts from the ESPRIT project OSSAD. Object types are defined for business resource (actors, data or documents) and business processes, and roles define their permissable states and constraints.

The use of business (or enterprise) models as contexts for conceptual models provide a particular world-view. Typically *business plans* [DO85] describe an organisation's operational and strategic structure through qualitative descriptions of mission, goals, objectives, critical success factors, market sectors, competitive and quality management strategies). These qualify the business services provided and organisational processing structure designed to carry out the business services, often occupying significantly sized documents. Business models abstract from this detail, describing concepts such as goal-

[5]This name was chosen as a result of communication with members of IFIP WG 8.1 Task Group FRISCO, a body commissioned to create a FRamework of Information Systems COncepts.

rooted organisational units, business services, activities, tasks, actors and actor roles (for processes) and resources (information and material).

[Ram94] provides an integrated executable specification framework both for business and IS modelling. At the strategic level of the business model, the problem solving for business services is carried out. At the operational level, concepts exist for activities and tasks, and information, material and actor resources. These are anchored into organisational units. Tasks are undertaken by a (primary) actor and possibly other actors. They consist of a partial order of elementary actions. An action is undertaken by a (primary) actor and possibly other actors. Actions apply to information/material objects. Figure 7 illustrates a part of an (operational) business model for the library domain.

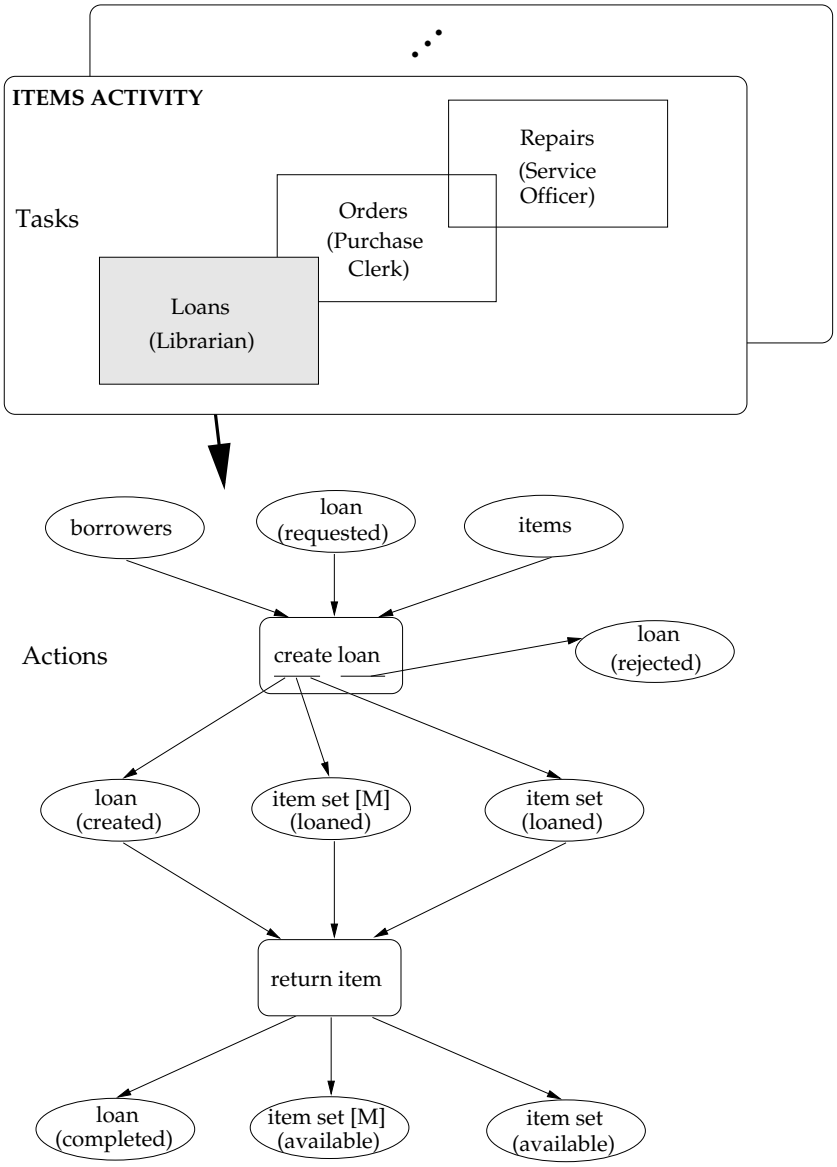

Figure 7: Operational business modelling for library example in [Ram94]

Figure 7 describes the action processing for the Items activity's Loans task which is undertaken by the

Librarian actor. Task modelling is CP-net based. The semantics of each action are as follows: an action is triggered when the precondition of all the input resources (actors and objects) being in the right state and a business rule (optional) is satisfied, the postcondition of resources being moved into the required states results. create loan is a composition for two exclusive actions which accept or reject the loan. A rule language augments action specifications although it is not as expressive as that in Hydra. In the example for instance, it is not possible to de-reference objects, items, within complex objects, loan. The business model is effectively developed through a process-centric integration, featuring the decomposition of an organisational processing structure. A number of views may be projected from the business model: data, process (note, not conceptually equivalent to structured process models), behaviour, object-based (with actions as methods), and the information systems in the broader sense (ISN) and information systems in the narrower sense (ISN) views motivated in [VR92].

The IS level is object-based and therefore adopts state-centric integration. Those object types which are selected to be computerised are refined at the IS level (associated actions having a computerised co-actor are defined as methods). The behavioural aspects of object types are defined through event precedences which trigger state transitions or perform retrievals. Event processing is also CP-net based with an ECA language. Three categories of object types are defined. Domain object types are normal IS object types, refined from computerised business information object types. View object types aggregate domain object types at human-to-computer interaction points (HCI) points which are derived through actions involving human and computerised co-actors. View object types trigger events on domain object types, therefore classifying application transactions. User object types simulate users, triggering events on view object types. Like the business level, a number of views may be projected: data, process (derived through inter-object event triggering, but not equivalent to structured process models), object-based, analysis and design.

Of contributory significance [Ram94] demonstrates that through the increased conceptualisation of business aspects, increased conceptualisation at the IS level is possible. As a result, application design mapping - an area of difficulty traditionally - is described. That is, through domain object type refinement, a business model is mapped to a conceptual IS model while through both view and user object type refinements, a business model is mapped to an external IS model. Two problems, both related to the business modelling technique, exist. The first is that the particular organisational processing structure is likely to restrict the suitability of the technique for any given business domain. The second, and more serious, is the basic treatment of an organisation's interaction with its external environment. Specifically, the interaction with the clients and other organisations is understood through the internal execution of tasks. Tasks form the only context from which actions are triggered. Events, including external events, are modelled at the IS level only.

To provide an abstraction from organisation processing structures, a new class of techniques, sometimes referred to as communication-based techniques, has been spawned by the work of [FL80] based on the speech-act theory of [Sea69]. The speech-act revisions of [Hab84] have been adopted by more recent such techniques, e.g. the Actor-Bank-Channel communication modelling technique [Die94] of the DEMO method. In it, the pattern of performative (state changing) conversations between actors is used to derive interaction structures for communication units known as essential[6] transactions. That is: the request for something, say a business service, results in an actagenic (or an action planning) conversation; followed by an essential action; and finally a factagenic (or fact generating) conversation, stating the results of the action. The actor initiating the actagenic conversation, i.e. initiator, is the same

[6] In DEMO the term essential qualifies non-computerisable processing despite the possibility of computerised support.

as the one terminating the factagenic conversation, i.e. executor. Conversely, the actor terminating the actagenic conversation is the same as the one initiating the factagenic conversation. For the execution of essential actions, actors generate plans involving communicative (e.g. information retrieval) actions from other actors in order to fulfill the essential action.

The same interaction structure applies to external transaction types, i.e. the initiator and the executor are external actors (in the environment), internal transaction types, i.e. the initiator and the executor are internal actors (in the business domain), and interface transaction types, i.e. the initiator is an external actor and the executor is an internal actor. A communication model is developed by building different transaction types into an interaction structure and including interstriction details (external data sources). A behaviour model is developed through the definition of execution and communication rules for performative conversations carried out by each actor. Significantly, different behaviour specifications may apply to different actors for the same object types.

The comparitive advantage of DEMO lies in its actor-centric basis for business process redesign. This formalises the otherwise ad-hoc socio-technical mechanisms in IS methods. Clearly, its most benefitial application lies in "fuzzy" business processing where the execution paths are uncertain, infrequent and highly fluctuating - i.e. ones which are difficult to document in operational procedure. This is well-recognised as a domain for CSCW. The benefit for the class of business processing addressed in this paper, given its overhead remains uncertain. Moreover, its impact at the detailed design level seems limited (compared to [Ram94]).

2.5 Summary

In this section, the findings of the assessment are generalised for the paradigms: structured process modelling, object-oriented modelling, behavioural-process modelling and business-oriented modelling.

Structured process modelling

Structured process modelling techniques provide a top-down process analysis where data flows and stores provide a process-centric integration with data modelling techniques. The simple and general set of concepts together with the feature of decomposition are applicable for different types of domains. At higher levels of analysis, models such as DFDs are easy to comprehend and can be adapted for business domains. A precise definition of what aspects of organisational processing structure should be modelled and how, is not available. That only basic aspects of business processes can be modelled is indicative of an insufficient business suitability. At the lowest level where the focus is on detailed design, pseudo-code is used to specify processes. The loosely-coupled integration structure, however, restricts expressive power. In general, the procedural style of conceptualisation forces a more imperative and less declaritive approach to specifications (than in behavioural process modelling techniques). Also process execution dependencies are not considered other than the process sequencing implicit in module design mapping. A major defficiency is the lack of a formal foundation, allowing ambiguities and inaccuracies in process models.

Object-oriented modelling

OOA/D techniques tightly-couple object structure and behaviour into an object model, citing as a motivation the "natural" occurence of objects in most types of domains. A state-centric integration is advocated principally, whereby an easily comprehensive object lifecycle (typically FSMs) is defined for each object type. In domains where object interaction is high, as tends to be the case in business domains, higher level contexts through state-centric (e.g. user cases and scenarios) or process-centric models (e.g. DFDs), are required. These extensions further necessitate an overall formal foundation, however little evidence of formal semantics was found in the surveyed OOA/D techniques. The resolution of difficult design decisions such as complex object aggregation and method classification through higher level models is therefore intuitive. Again, generality inhibits business suitability, although a large class of business rules can be specified through object lifecycles. In this regard, a conceptual specification language is required to augment ECA specifications.

Behavioural process modelling

The behavioural aspect of process models fundamentally concerns execution dependencies between processes, i.e. control flow. This includes the sequence, repetition, parallelism and synchronisation of processes. Despite generality, these features allow greater semantics of business processes to be captured. Traditionally, given their precise and graphically communicable operational semantics, Petri nets have been used to develop behavioural process modelling techniques. As such, model validation through execution is also possible. The inclusion of states in process models - note, still process-centric - provides a declaritive approach to process specifications, and therefore to a further class of business rules. However, the impact on the graphical representation is problematic, with process models becoming cluttered for even basic specifications. Strategies to alleviate this include the use of abstraction/decomposition and the use of conceptual specification languages for detailed specifications. The use of formalisms like Petri nets does not guarantee a formal foundation; formal semantics still have to be defined. Algebraic systems (such as Process Algebra) provide an alternate mechanism for the definition of formal semantics.

Business-oriented modelling

Business-oriented techniques specialise conceptual modelling so that a precisely defined business suitability is achieved. For this, an alignment of concepts with organisational processing structure and a provision of colloborative processing are required. Unless otherwise addressed, the quality of conceptualisation, expressive power, comprehensibility and formal foundation are inherited from the "underlying" classical techniques. Some techniques cater for a divergent set of business world-views through a basic, but not necessarily complete set of business constructs. Other techniques prescibe a particular world-view - a business model. In integrated specification environments, IS models are refined from business models and so the formal semantics of the IS model should relate closely to that of the business model. Also, the structural dependency of IS models on business models can occur, therefore violating the Conceptualisation principle. Communication-based techniques focus on organisational communication (a speech-act synthesis of actor communication). This leads to a more essential insight into business process redesign.

3 Business suitability principles

In this section, a diagnosis of the business suitability of workflow modelling is presented. Recall from section 1, attention is confined to mission-critical business processing which is amenable to strict operational procedure. Both business services and business processes are essential components of this business processing. Although highly inter-related, they require (as diagnosed) a careful modelling distinction.

A discussion of the common problems follows, together with an elicitation of business suitability to alleviate these. The list of principles are:

- Organisational embedding (section 3.1)
- Scenario validation (section 3.2)
- Service information hiding (section 3.3)
- Cognitive sufficiency (section 3.4)
- Execution resilience (section 3.5)

3.1 Organisational embedding

The first problem relates to the relationship between business models and conceptual models. On the one hand, the adaption of classical techniques with some business-oriented constructs leaves room for an arbitrary relationship. On the other hand, the incorporation of business models might well lead to the situation where organisation processing structure prescribes the essential structure of the conceptual model; e.g. when an IS model is decomposed hierarchically from a business model. This, of course, is a violation of the Conceptualisation Principle. Moreover, it may lead to an inflexibility in IS design

In [Ram94] for example, it was observed that computerised task actions are mapped directly into object methods. While an IS design specification may be decomposed legitimately in this fashion, there are other situations where IS modules are designed from a composition of tasks, possibly drawn from multiple organisations. The point is that such an abstraction should be reflected at the IS level, and not the business level since no such (single) business task exists. The task composition serves an IS design purpose only.

To avoid these problems, the following principle is proposed:

Principle – *Organisational Embedding*
A technique should embed all concepts in a conceptual model, directly or indirectly, but without redundancy, into organisational elements. □

Note, the principle does not prescribe the use of business models. Rather, it states that any modelling concepts be backtracked to organisational elements (whatever form of their definition). In above example, the dependency of IS models on business models is replaced by an inter-dependency. That is, IS and business models are permitted their own modelling autonomies, and a consistency between the two is required. The consistency does not preclude mechanisms such as networked decompositions. Therefore, business task compositions are permitted for IS process design.

3.2 Scenario validation

The second problem stems from the observation that all techniques deal, at best, only partially with the validation of models. Validation is concerned with ensuring that a conceptual model is indeed a model of a business domain. Beyond the validation of partial models, no support was evident for an organisationally embedded validation mechanism which "cuts across" the partial models, drawing their concepts into a unified cognition. Both the suitability and comprehensibility of a technique are important factors in model validation, and it was evident in the survey that different integration strategies offered different advantages.

In classical techniques it was observed that triggering sources such as external events are used to trace the execution paths in models. In process-oriented techniques the resultant process execution sequence is determined implicitly in structural process models and explicitly in behavioural process models. The validation of execution paths is also advanced in behaviour process models through synchronisation and decision constructs. In OOA/D techniques, process sequences apply when object interactivity is high while object state sequences (in accordance with object lifecycles) apply when object interactivity is low. In business-oriented techniques, it was quite clear that organisational processing structures or organisation communication provide focal units for business processing.

A common theme for model validation is the notion of event. At discrete points within a timeline, it can be seen that an event triggers actors to execute business processes, through which information may be accessed, and further events, possibly invoking some processing in the environment (i.e. outside the business domain), may result. Intrinsic to the initial event is some defined purpose which motivates an execution path all the way to the final event. The final event signifies the (logical) termination of processing; i.e. representing the organisation's recognition that no further processing should proceed. Such an execution path is referred to as a *scenario*. For the purposes of model validation therefore, the following principle is proposed:

Principle – *Scenario Validation*
A technique should provide an explicit notion of scenario for model validation. □

Note, the principle does not exclude other forms of model validation, e.g. the validation of constraints in a data model. Rather, it requires that scenarios, amongst other possible forms of validation, be supported by a technique. At the same time, is should be pointed out that since a scenario provides a systematic means by which execution paths are traced, it should lead to different parts of conceptual submodels, from which further validation may then follow. Hence, scenarios should provide a more effective completeness check than partial model checks.

Of issue is the organisational embedding of a scenario. An obvious choice is the notion of a *business transaction*, drawn from a Macroeconomics perspective of organisations [BM91]. Business transactions concern the exchange of (goods and) services using business transactions as the fundamental (accounting) unit. Clearly, the determination of a business transaction cost requires an understanding of the processing undertaken. It may be simple involving a low interactivity: a small set events and business processes. Or it may be complex involving a high-interactivity over a "long" temporal duration: a large set of events and business processes, typically involving multiple organisations.

At a first glance, it may seem that the choice of business service over business transaction as a scenario is arbitrary. Certainly, both terms tend to be used interchangeably in IS research. A cause-effect

distinction however, leads to the preference of business transactions. That is, business services are a cause reflecting, more than anything else, client requirements of the organisation whereas business transactions are the effect; in effect, the scenario which results. In this sense, the concept of business transaction as an organisational artefact may be unified with the concept of workflow as an IS design and implementation artefact, i.e. *business transaction workflow*. In stating this, it should be understood that that class of business transactions involving uncertain, indeed "fuzzy" execution paths, has been excluded from the scope of this paper (section 1).

3.3 Service information hiding

Following from the preceding discussion, the third problem lies in the relationship between events, business processes and business services. Recall, events trigger process execution for some ultimate intention. Moreover, a number of events may be related to the same intention. In a business sense, intentions are denoted by business services.

In classical modelling techniques, no explicit service concept exists. Business services are therefore dealt with in an arbitrary fashion. In process modelling techniques for instance, services are qualified (informally) in event descriptions and therefore directly trigger processes (see examples in [You89]). Similarly in OOA/D techniques, events directly trigger methods in object types (recall Figure 3). Since business-oriented techniques adapt classical techniques, their treatment of business services is also arbitrary. For example in [Ram94], business services (functions) are related to business processes (tasks), but this is no more than a reference; i.e. triggering relates directly to business processes.

The common problem in these approaches is the *direct* triggering of processes (or methods) given the context of triggering. From the point of view of the environment or from different parts of an organisation, the actual business processes triggered for some business service request are inconsequential for the formulation of the request. That is, the request is issued for a business service and as a result some internal mechanism is used to determine what action to take. The implication is that when processes are reengineered, the actual request is not affected.

It may be recognised that the above paragraph reflects the *Information Hiding* principle for (software) module design which requires that internal aspects of modules be insulated from other modules. When applied to a technique's treatment of business service triggering, it can be seen that a service concept should explicitly be supported to coordinate the execution of number of processes and insulate them from service requests. Or in a more general form, the following principle results:

Principle – *Service Information Hiding*

A technique should allow the formulation of service requests to be independent of their actual processing. □

3.4 Cognitive sufficiency

The fourth problem is fairly general in its diagnosis. It addresses the missing constructs in techniques which yield collectively a sufficient *cognition*[7] of a business processing model, i.e. business transaction model. Fundamental areas of variance are discussed below.

[7] A pragmatic rather than a theoretical diagnosis is presented. This is based on the comparative differences in the assessed techniques and their advantages on model cognition.

Process modelling cognition

The process modelling in most techniques was observed to incorporate essentially either structural (data flows) or behavioural (control flows) aspects, but not both. In BNM, DFDs and PrT-nets are both used, but at different levels of abstraction. In [Ram94], a process model view is derived from process behaviour, and so it is not structural. Specifically, the data flows are a set of attribute values rather than identifiable *containers* of those values, while data stores are object states rather than identifiable repositories of data. In Hydra, a 'middle ground' is apparent where process behaviour is combined with some structural aspects: process decomposition and a notion of data stores (temporary storage). At the other extreme, in Structured Analysis, process behaviour is altogether omitted from DFDs except for process sequencing (only) defined as part of software design mapping.

Structural and behavioural aspects of process models serve distinctly different aspects of analysis. Data flows, stores and transformations from the structural aspect lead to an understanding of data dependencies, from which business processes and data stores may be redesigned. As examples, common patterns of data flow in different process types indicate the need for process reclassification, while dependencies on a number of data stores for a data transformation function indicate the need for data relocation. Process sequence, repetition, parallelism and synchronisation from the behavioural aspect, on the other hand, lead to an understanding of process dependencies from which the problems of deadlock, livelock and starvation may be identified. As a result, execution paths may be optimised or processes themselves may be reclassified. Interestingly, the strategies defined in [Ham90] indicate that a combination of both - simultaneously - is required for business processing reengineering. This indicates that both are required to sufficiently capture and understand a business transaction's semantics.

HCI

Another area of cognitive insufficiency is the ad-hoc conceptualisation of HCIs (recall human to computer interactions). For HCI, two issues are apparent [DMHB90], namely the point (process) at which a HCI is required and the *dialogue* which determines the screen/form design. Under classical analysis, as evident from the discussions of process and object modelling techniques, HCIs are determined as part of systems design where application services, which typically encapsulate one or more screens/forms, are determined intuitively from conceptual models. This situation, as remarked more than once, leads to a waterfall since the backtracking of specifications may be difficult. In [Ram94], it was seen that business model semantics, in this case the actor types involved, can be used to derive HCI points. [Die94] synthesises actor-actor communication so that dialogues go beyond HCI where it is argued, systems design including HCI, may be better understood.

Temporal aspects

Finally, the treatment of temporal aspects is partial despite the (obvious) impact of time in business domains. In Structured Analysis and other DFD techniques, a "clock" symbol is introduced to indicate time triggers on processes, e.g. end of the week. This however is only a comprehensibility feature. In [Ram94], time triggers are incorporated into the event specification language for object types, thereby extending the concern to expressive power. Further to expressive power, the need for time functions and variables was indicated in the discussions of BNM and Hydra (recall the preconditions of BC1). A

survey of techniques (including TEMPORA) dealing with time [TL91], demonstrates the incorporation of time on action preconditions, and relatedly dynamic constraints on object type specifications.

It is clear that these incorporations of temporal aspects relate to the specification of process preconditions. Of course, time should also be important in postconditions, e.g. as motivated by event scheduling requirements in [Bri90]. This of course further qualifies process dependency since a number of processes may be required to execute at the at some time simultaneously (parallelism) or within a time duration of each other (sequence). Also a process may execute repeatedly within a certain time period (repetition), until some condition is satisfied. It is interesting to note that some model-based formal specification languages e.g. [Zav86], specification languages e.g. Real-Time Process Algebra [BB91], and Petri net based approaches e.g. ExSpect [HSV89], all address temporal aspects in process specifications. Indeed, the traditional domain for this consideration has been in real-time systems. The need of temporal specification, is nonetheless, evident in business domains. For example in law courts, matters are scheduled and adjourned at designated times, and in doing so, the availability of required documentation is requested (to outside organisations and parties) within a certain duration prior to the court hearing.

Principle – *Cognitive Sufficiency*

A technique should provide a sufficient cognition of a model such that the need for fundamental business process execution assumptions is eliminated. □

3.5 Execution resilience

The fifth problem relates to error handling at the conceptual level. Of course at the conceptual level, database constraints and process pre- and post-conditions define an error free IS state. Moreover, specifications may be verified to eliminate erroneous specifications. However, *operational* errors can still occur beyond the control of an IS. For example, in a library domain, the existence of an overdue item (over different categories of duration) is an operational error(s) caused by a borrower. Similarly, a missing enrollment confirmation in the required time since an initial enrollment, is caused by a student. Finally, a system crash is one of several examples of operational errors resulting in non-deterministic processing failures.

A sound conceptualisation of business transaction semantics requires the the inclusion of operational error handling. However, no evidence in the assessed techniques indicated an explicit consideration of this. In most cases, an expressive power is available for dealing with errors like the first two examples. In process-centric techniques for example, error handling processes may be defined to scan a database for problematic object states (overdue item, unconfirmed enrollment). In state-centric techniques, an error handling process may be fired as a result of an object's transition into a problematic state.

The treatment of non-deterministic failures has recently become the subject of workflow implementation specifications. In particular, the traditional transaction model (more recently and comprehensively described in [GR93]) with its ACID properties (atomicity, consistency, isolation and consistency) has been extended for workflow execution semantics. Under the traditional model, a transaction binds a set of database operations into an atomic unit of execution. Following the requirement of *failure atomicity*, a transaction's changes to a database(s) are *committed* if the execution is successful or *rolled-back* if not. Following the requirement of *execution-atomicity*, the concurrent execution of transactions should have the same effect as if they were executed in a *serialisable* order.

Workflows are more complex structures than traditional transactions, and it is unacceptable that the failure of any one of its tasks results in the rollback of the entire workflow. The application of transaction models to workflows, typically through a relaxation of ACID, are surveyed in [Kim94] (pp.596-598). Under an ACID relaxation, a failure atomicity is defined for each task. This involves either an *undo*, possibly through the execution of some other task, i.e. a *compensation*, or a *redo*, possibly through a contingent task execution. Committed tasks are therefore not re-executed. Committed tasks, however, may update data objects and release locks, allowing other tasks and transactions to "see" the updates. Hence the need for compensations, which either logically undo the updates or provide some notification of update invalidity. Of course, compensating tasks should not, themselves, fail.

The lack of explicit support for operational error handling is indicative of the traditional IS development approach which prefers this treatment at the the implementation level. A certain amount of this detail is, afterall, implementation-oriented (e.g. checking for a DBMS-defined deadlock error code after an update to determine whether a retry should be issued). Yet it should be recognised that operational error handling is an inherent part of a business transaction's semantics. Where the previously described conceptualisation aspects relate to the *Cognitive Sufficiency* of normal execution semantics, the conceptualisation of operational error handling relates to *execution resilience* semantics. As repeatedly discussed, the partial conceptualisation of some aspects and not others, can distort the overall understanding of the business transaction semantics.

Significantly, the support of model execution in a large number of techniques, means that execution resilience can actually be validated and "tested" at the conceptual level. Given the disjointness (and complexity) of operational error handling - best presented as a separate layer to preserve a model's general cognition - model execution is considered critical. Therefore the following principle results:

Principle – *Execution Resilience*
A technique should support the handling of operational errors, so that a resilient execution of the conceptual model results. □

Ultimately, such support addresses a workflow's execution resilience. This of course does not guarantee execution resilience since operational errors, e.g. crashes, can (and do) happen.

4 Epilogue

General problem

Conceptual modelling techniques haven proven to be invaluable for early and critical phases of analysis and design. When "bundled" into well-formed IS methods, a navigation from organisational analysis to implementation specifications is possible. For an essential problem-solving insight into different aspects of business domains, techniques should demonstrate a conceptualisation, expressive power, comprehensibility, formal foundation and business suitability. In the last of these aspects, the absence of a universal organisational theory has meant that techniques can only increase in technical effectiveness as newer insights are obtained from practical experience, and as the functionality of IS platforms expands. This makes it difficult to assess the adequacy of techniques to support a sound conceptualisation of business domains.

Focus of paper

In this paper, the business suitability of workflow modelling was diagnosed for that class of business processing which is mission-critical in nature and which is amenable to strict operational procedure. The approach taken was two-fold. First, the general capabilities of techniques characteristic of well-known paradigms were assessed using the general requirements. Second, the extent to which workflows scale out to a business suitability - "constructed" from the business processing inter-relationship of business service *and* business processes - this combination providing a new insight in our opinion - was diagnosed. As a result, five new business suitability principles were formulated.

General assessment

The findings of the general assessment were as follows. Classical techniques, i.e. process (structured and behavioural) and object modelling, offer general concepts and features which are applicable to several types of domains. Where they differ is in the cognitive dependencies of their integration strategies. Process-centricity emphasises data flow transformations (structured) and process control dependencies (behavioural). Both are useful for describing the flow of business processing. State-centricity emphasises data object states and the transitions between them. This is useful for focusing event (signaling/messaging) and their impacts on business processing. Since an object lifecycle deals only with one object type, a number of OOA/D techniques also incorporate some form of process-centricity. In general, the need for a trade-off between comprehensibility and expressive power was conspicuous throughout the assessment. Techniques biasing the former (e.g. DFDs) tend to be weak in the latter (e.g. pseudo-code), conversely those biasing the latter (e.g. Petri net based approaches) tend to be weak in the former (e.g. cluttered diagrams). Also an adequate formal foundation, in particular formal semantics, was notably absent in most techniques. This raises ambiguities in the application of techniques, and in turn makes it impossible to prove certain properties about their models. In this regard, the absence of correctness checking in workflow specifications (noted in section 1) is striking.

Business-oriented modelling adapt combinations of classical techniques within the context of some inter-subjective business world-view. This may be partial, i.e. the incorporation of some (key) business concepts, or it may be complete, i.e. the incorporation of a business model. The use of essential business modelling concepts such as business services, processes (of various specialisations), actors and actor roles, organisational units and resources (material and information) has become popular, particularly through integrated specification environments such as CAiSE tools and requirements engineering methods. Several approaches adopt a decomposition of organisational processing structure. In so doing, a more automatic application design mapping is demonstrated compared to classical techniques. Recently proposed communication-based techniques abstract from organisational processing structure and analyse human speech-acts as a basis for understanding business processing (amongst other forms of processing). This allows more detailed aspects of workflow interaction to be understood, however the extent of its applicability to the class of business processing we considered was not clear.

Business suitability principles

The detailed diagnosis of business suitability led to the identification of the following problems, for which associated principles were defined:

- A hierarchical decomposition from a business model to an IS model can result in a violation of the Conceptualisation Principle whereby an essential structure of the IS model is prescribed. For example, a hierarchical decomposition does not allow IS processes to be composed out of distinct organisational processing structures, say from more than one organisation. To provide techniques with their own abstraction autonomy, hierarchical decomposition is replaced by the *Organisational Embedding Principle*. It simply requires a link of any given IS concept, whether direct or indirect, into organisational elements. As a result, the inter-dependency between the business and IS levels does not impact the conceptualisation at these levels.

- Inherent in business processing is a scenario, i.e. an execution sequence which occurs over a time duration for some particular intention. Scenarios are important since they provide a domain semantic interpretation, i.e. validation, of a model. Support for scenarios *per se*, is absent in classical techniques, and present in some form, through either organisational processing structures and organisation communication, in business-oriented techniques. The need for explicit support was defined through the *Scenario Validation Principle*. The notion of business transactions, recognised within Macroeconomics theory, further qualified the business service and business process inter-relationship as a scenario. That is, the triggering of a business service for a request, and any subsequent triggering, including those on business processes and business services resulting in (some defined) satisfaction of the request, constitutes a business transaction; an essential scenario concept.

- Related to the above, all techniques directly or potentially lead the knowledge of "internal" processing required during the formulation of a service request. This, of course, violates the well-known Information Hiding Principle. To alleviate this, the *Service Information Hiding Principle* was defined. As such, service requests are completely insulated from subsequent triggering. Put simply, services coordinate process execution.

- General omissions were identified which restricted the cognition of a business transaction model. These included: the support of either structured (e.g. data flow) or behavioural (e.g. control flow) aspects for process modelling, but not both; the ad-hoc conceptualisation of HCIs; and finally (and surprisingly) the partial support for temporal aspects (e.g. not considered in process postconditions). When not supported by a technique, erroneous assumptions can result. The *Cognitive Sufficiency Principle* was defined to ensure that such fundamental assumptions are eliminated.

- Operational errors are those which occur during IS operation, but outside control of the IS. Although most techniques have an expressive power to support some types of operational error handling specifications, and although the handling of non-deterministic operational failures is considered crucial in workflow specifications, no explicit support is provided by the techniques. When specified at the implementation, level, the overall context of business processing semantics may be lost. Furthermore, any model validation through execution is partial, bypassing operational errors. The *Execution Resilience Principle* was defined to ensure that operational errors are specified at the conceptual level, so that a model may be "tested" for execution resilience. This addresses the execution resilience of an IS ultimately.

Conclusion

To conclude, the business suitability principles address an important yet not often salient aspect of IS conceptual modelling. Of course, we focussed the diagnosis on a particular aspect of business domains, namely a particular class of business processing, characterised in a particular way, and intended for a particular implementation model - namely that based on the workflow concept. Undoubtedly, further considerations of this and other aspects of business domains are still required. Importantly, our characterisation of business processing is well-recognised, and the conceptual level of diagnosis imparts relevance to traditional implementation models as well. Moreover by describing rather than prescribing the principles, we seek to position our work more effectively with other such developments.

The major benefit lies in assessing techniques, whereby immediate attention may be drawn to the general areas of deficiency we encountered. To overcome these and other deficiencies, it should now be clear that enhancing a technique requires a simultaneous consideration of all the conceptual modelling requirements. Given the large number of useful contributions in the field, we advocate a synthetic rather than analytic approach to developing techniques.

Another benefit lies in the insight gained for workflows involving global access (e.g. via the Web), inter-organisational collaboration and reuse of application services. A clear separation between business service, business process and business transaction not only improves the conceptual modelling but may well have far reaching implications for workflow specification languages and, ultimately, for (middleware) description languages of open distributed architectures.

Future work will deal with the development of a *kernel* of concepts and features for workflow modelling along the lines of the principles.

References

[AMP94] A.I. Anton, W.M. McCraken, and C. Potts. Goal decomposition and scenario analysis in business process reengineering. In G. Wijers, S. Brinkkemper, and I. Wasserman, editors, *Proceedings of the Sixth International Conference CAiSE'94 on Advanced Information Systems Engineering*, volume 811 of *Lecture Notes in Computer Science*, pages 94–104, Utrecht, The Netherlands, June 1994. Springer-Verlag.

[AW91] D.E. Avison and A.T. Wood-Harper. Information Systems Development Research: An Exploration of Ideas in Practice. *The Computer Journal*, 34(2):98–112, April 1991.

[BB91] J.C.M. Baeten and J.A. Bergstra. Real Time Process Algebra. *Formal Aspects of Computing*, 3:142–188, 1991.

[BB95] A.T. Berztiss and J.A. Bubenko. A software process model for business reengineering. In *Proceedings of Information Systems Development for Decentralized Organizations (ISDO95), an IFIP 8.1 Working Conference*, pages 184–200, Trondheim, Norway, August 1995. Chapman & Hall.

[BCN92] C. Batini, S. Ceri, and S.B. Navathe. *Conceptual Database Design - An Entity-Relationship Approach*. Benjamin Cummings, Redwood City, California, 1992.

[BM91] W. Boyes and M. Melvin. *Macroeconomics*. Houghton Mifflin, Boston, Massachusetts, 1991.

[Boo91] G. Booch. *Object-Oriented Design with Applications*. Benjamin Cummings, Redwood City, California, 1991.

[Bot89] P.W.G. Bots. *An Environment to Support Problem Solving*. PhD thesis, Delft University of Technology, Delft, The Netherlands, 1989.

[Bri90] S. Brinkkemper. *Formalisation of Information Systems Modelling*. PhD thesis, University of Nijmegen, Nijmegen, The Netherlands, 1990.

[Bro87] F.P. Brooks Jr. No silver bullet: essence and accidents of software engineering. *IEEE Computer*, 20(4):10–19, April 1987.

[BS87] D. Benyon and S. Skidmore. Towards a Tool Kit for the Systems Analyst. *The Computer Journal*, 30(1):2–7, 1987.

[BW89] P. D. Bruza and Th.P. van der Weide. The semantics of data flow diagrams. In N. Prakash, editor, *Proceedings of the International Conference on Management of Data (CISMOD)*, pages 66–78, Hyderabad, India, 1989. McGraw-Hill Publishing Company.

[CN90] S. Chakravarthy and S. Nesson. Making an object-oriented DBMS active: design implementation and evaluation of a prototype. In *Proceedings of the International Conference on Extended Database Technology (EDBT)*, Venice, Italy, April 1990.

[CP96] P.N. Creasy and H.A. Proper. A Generic Model for 3-Dimensional Conceptual Modelling. *Data & Knowledge Engineering*, 20(2):119–162, 1996.

[CY90] P. Coad and E. Yourdon. *Object-Oriented Analysis*. Yourdon Press, New York, New York, 1990.

[Dav90] A.M. Davis. *Software Requirements: Analysis & Specification*. Prentice-Hall, Englewood Cliffs, New Jersey, 1990.

[DB91] R.C.J. Dur and P.W.G. Bots. Dynamic Modelling of Organizations Using Task/Actor Simulation. In R.L. Crosslin and H.G. Sol, editors, *Proceedings of the Second International Working Conference on Dynamic Modelling of Information Systems*, pages 49–71, Amsterdam, The Netherlands, 1991. Elsevier Science Publishers.

[DeM78] T. DeMarco. *Structured Analysis and System Specification*. Prentice-Hall, Englewood Cliffs, New Jersey, 1978.

[Die94] J. Dietz. Business modelling for business redesign. In *Proceedings of the 27th International Conference on System Sciences*, pages 723–732, Honolulu, Hawaii, August 1994.

[DMHB90] E.J.T. van Dinter, M.P.W. Martens, A.H.M. ter Hofstede, and S. Brinkkemper. A Support Tool for Integrated Dialogue Specification. In R. Norman and R. van Gendt, editors, *Proceedings of the Fourth International Workshop on CASE*, pages 290–301, Irvine, California, December 1990. IEEE Computer Science Press.

[DO85] G.B. Davis and M.H. Olson. *Management Information Systems: Conceptual Foundations, Structure and Development*. McGraw-Hill, New York, New York, 1985.

[DP95] V. De Antonellis and B. Pernici. Reusing specifications through refinement levels. *Data & Knowledge Engineering*, 15(2):109–133, April 1995.

[DZ81] V. De Antonellis and B. Zonta. Modelling Events in Database Application Design. In C. Zanioli and C. Delobel, editors, *Proceedings of the 7th VLDB Conference*, pages 23–31, Cannes, France, August 1981.

[EGH+92] G. Engels, M. Gogolla, U. Hohenstein, K. Hülsmann, P. Löhr-Richter, G. Saake, and H-D. Ehrich. Conceptual modelling of database applications using an extended ER model. *Data & Knowledge Engineering*, 9(4):157–204, 1992.

[EJW95] D.W. Embley, R.B. Jackson, and S.N. Woodfield. OO Systems Analysis: Is It or Isn't It? *IEEE Software*, 12(3):19–33, July 1995.

[FHL+98] E.D. Falkenberg, W. Hesse, P. Lindgreen, B.E. Nilsson, J.L.H. Oei, C. Rolland, R.K. Stamper, F.J.M. Van Assche, A.A. Verrijn-Stuart, and K. Voss, editors. *A Framework of Information Systems Concepts*. IFIP WG 8.1 Task Group FRISCO, 1998. ISBN 3-901-88201-4

[Fir91] D. Firesmith. Structured analysis and object-oriented development are not compatible. *ACM Ada Letters*, 11(9):56–66, 1991.

[FL80] F. Flores and J.J. Ludlow. Doing and Speaking in the Office. In *Decision Support Systems: Issues and Challenges*. Pergamon, 1980.

[FYW94] G. Fitzpatrick, Y. Yang, and J. Welsh. Supporting Cooperative Work Processes - A Survey of Systems and Issues. Technical Report 304, Department of Computer Science, University of Queensland, Brisbane, Australia, June 1994.

[Gen87] H. Genrich. Predicate/Transition Nets. In W. Brauer, W. Reisig, and G. Rozenberg, editors, *Petri Nets: Central Models and Their Properties, Advances in Petri Nets 1986 Part I*, volume 254 of *Lecture Notes in Computer Science*, pages 207–247. Springer-Verlag, Berlin, Germany, 1987.

[GHS95] D. Georgakopoulos, M. Hornick, and A. Sheth. An Overview of Workflow Management: From Process Modelling to Workflow Automation Infrastructure. *Distributed and Parallel Databases*, 3(2):119–153, April 1995.

[GR93] J. Gray and A. Reuter, editors. *Transaction Processing: Concepts and Techniques*. Morgan Kaufmann, San Mateo, California, 1993.

[GS86] C. Gane and T. Sarson. *Structured System Analysis: Tools and techniques*. IST Databooks. MacDonald Douglas Corporation, St. Louis, 1986.

[Hab84] J. Habermas. *The Theory for Communicative Action: Reason and Rationalization of Society*, volume 1. Boston Beacon Press, Boston, Massachusetts, 1984.

[Ham90] M. Hammer. Re-engineering work: don't automate, obliterate. *Harvard Business Review*, 68(4):104–112, April 1990.

[HL91] X. He and J.A.N. Lee. A Methodology for Constructing Predicate Transition Net Specifications. *Software Practice & Experience*, 21(8):845–875, August 1991.

[HN93] A.H.M. ter Hofstede and E.R. Nieuwland. Task structure semantics through process algebra. *Software Engineering Journal*, 8(1):14–20, January 1993.

[Hof93] A.H.M. ter Hofstede. *Information Modelling in Data Intensive Domains*. PhD thesis, University of Nijmegen, Nijmegen, The Netherlands, 1993.

[HPW93] A.H.M. ter Hofstede, H.A. Proper, and Th.P. van der Weide. Formal definition of a conceptual language for the description and manipulation of information models. *Information Systems*, 18(7):489–523, October 1993.

[HSV89] K.M. van Hee, L.J. Somers, and M. Voorhoeve. Executable Specifications for Distributed Information Systems. In E.D. Falkenberg and P. Lindgreen, editors, *Information System Concepts: An In-depth Analysis*, pages 139–156. North-Holland/IFIP, Amsterdam, The Netherlands, 1989.

[HU79] J.E. Hopcroft and J.D. Ullman. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, Reading, Massachusetts, 1979.

[HW93] A.H.M. ter Hofstede and Th.P. van der Weide. Expressiveness in conceptual data modelling. *Data & Knowledge Engineering*, 10(1):65–100, February 1993.

[ISO87] *Information processing systems – Concepts and Terminology for the Conceptual Schema and the Information Base*, 1987. ISO/TR 9007:1987.
`http://www.iso.org`

[JCJO92] I. Jacobson, M. Christerson, M. Jonsson, and P. van Overgaard. *OO Software Engineering, A Use Case Driven Approach*. Addison-Wesley, Reading, Massachusets, 1992.

[Jen91] K. Jensen. Coloured Petri Nets: A High Level Language for System Design and Analysis. In G. Rozenberg, editor, *Advances in Petri Nets 1990*, volume 483 of *Lecture Notes in Computer Science*, pages 342–416, Berlin, Germany, 1991. Springer-Verlag.

[JOS93] P. Jaeschke, A. Oberweis, and W. Stucky. Deriving Complex Structured Object Types for Business Process Modelling. In P. Loucopoulos, editor, *Proceedings of the 13th International Conference on the Entity-Relationship Approach*, Lecture Notes in Computer Science, pages 28–45, Manchester, United Kingdom, December 1993. Springer-Verlag.

[Kim94] W. Kim, editor. *Modern Database Systems: The Object Model, Interoperability and Beyond*. Addison-Wesley, Reading, Massachusetts, 1994.

[Kun93] D.C. Kung. The Behaviour Network Model for Conceptual Information Modelling. *Information Systems*, 18(1):1–21, 1993.

[LGN81] M. Lundeberg, G. Goldkuhl, and A. Nilsson. *Information Systems Development - A Systematic Approach*. Prentice-Hall, Englewood Cliffs, New Jersey, 1981.

[LK95] P. Loucopoulos and E. Kavakli. Enterprise modelling and the teleological process design and database design. *International Journal of Cooperative Information Systems*, 4(1):45–79, January 1995.

[LMS+91] P. Loucopoulos, P. McBrien, F. Schumaker, B. Theodoulidis, V. Kopanas, and B. Wangler. Integrating database technology, rule based and temporal reasoning for effective information systems. *Journal of Information Systems*, 1:129–152, February 1991.

[McC92] S. McCready. There is more than one kind of work-flow software. *Computerworld*, November 1992.

[ML83] J.L. Malouin and M. Laundry. The Mirage of Universal Methods in Systems Design. *Journal of Applied Systems Analysis*, 10:47–62, 1983.

[MMNR90] V.J. Mercurio, B.F. Meyers, A.M. Nisbet, and G. Radin. AD/CYCLE strategy and architecture. *IBM Systems Journal*, 28:170–187, 1990.

[OHM+88] T.W. Olle, J. Hagelstein, I.G. Macdonald, C. Rolland, H.G. Sol, F.J.M. van Assche, and A.A. Verrijn-Stuart. *Information Systems Methodologies: A Framework for Understanding*. Addison-Wesley, Reading, Massachusetts, USA, 1988. ISBN 0-201-54443-1

[OS93] A.L. Opdahl and G. Sindre. A taxonomy for real-world modelling concepts. *Information Systems*, 19(3):229–241, 1993.

[PD94] L.F. Pollacia and L.M.L. Delcambre. The object flow model: a formal framework for describing dynamic construction, destruction and interaction of complex objects. In P. Loucopoulos, editor, *Proceedings of the 14th International Conference on the Entity Relationship Approach*, Lecture Notes in Computer Science, pages 1–12, Manchester, United Kingdom, December 1994. Springer-Verlag.

[Pet81] J.L. Peterson. *Petri Net Theory and the Modelling of Systems*. Prentice-Hall, Englewood Cliffs, New Jersey, 1981.

[Ram94] G.J. Ramackers. *Integrated Object Modelling, an Executable Specification Framework for Business Analysis and Information System Design*. PhD thesis, University of Leiden, Leiden, The Netherlands, 1994.

[RBP+91] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, and W. Lorenson. *Object-Oriented Modeling and Design*. Prentice-Hall, Englewood Cliffs, New Jersey, 1991.

[RD82] G. Richter and R. Durchholz. IML-Inscribed High-Level Petri Nets. In T.W. Olle, H.G. Sol, and A.A. Verrijn-Stuart, editors, *Information Systems Design Methodologies: A Comparative Review*, pages 335–368. North-Holland/IFIP WG8.1, Amsterdam, The Netherlands, EU, 1982.

[Rei85] W. Reisig. *Petri Nets: An Introduction*. EATCS Monographs on Theoretical Computer Science. Springer-Verlag, Berlin, Germany, 1985.

[Rod91] T. Rodden. A survey of CSCW systems. *Interacting with Computers*, 3(3):319–353, 1991.

[RV85] G. Richter and K. Voss. Towards a comprehensive office model integration information and resources. In *Advances in Petri Nets*, volume 222 of *Lecture Notes in Computer Science*, pages 401–417. Springer-Verlag, 1985.

[Sea69] J.R. Searle. *Speech Acts: An Essay in the Philosophy of Language*. Cambridge University Press, Cambridge, United Kingdom, 1969.

[Sho91] P. Shoval. An integrated methodology for functional analysis, process design and database design. *Information Systems*, 16(1):49–64, 1991.

[Shu91] K. Shumate. Structured analysis and object-oriented design are compatible. *ACM Ada Letters*, 9(4):78–90, 1991.

[SI90] P. Sully and D. Ince. The synthesis of object-oriented designs from the products of structured analysis. In P.A.V. Hall, editor, *Proceedings of Software Engineering 90*, pages 404–432, 1990.

[SJ91] S.C. Solsi and E.L. Jones. Simple yet complete heuristics for transforming data flow diagrams into Booch style diagrams. *ACM Ada Letters*, 9(2):115–127, March 1991.

[SK86] A. Sølvberg and C.H. Kung. On Structural and Behavioural Modelling of Reality. In T.B. Steel Jr and R. Meersman, editors, *Database Semantics (DS-1)*, pages 145–171. North-Holland/IFIP, Amsterdam, The Netherlands, 1986.

[SM88] S. Shlaer and S.J. Mellor. *Object-Oriented Systems Analysis: Modeling the World in Data*. Yourdon Press, New York, New York, 1988.

[TL91] C.I. Theodoulidis and P. Loucopoulos. The Time Dimension in Conceptual Modelling. *Information Systems*, 16(3):273–300, 1991.

[Ver93] P.A.C. Verkoulen. *Integrated Information System Design - An Approach Based on Object-Oriented Concepts and Petri Nets*. PhD thesis, Eindhoven University of Technology, Eindhoven, The Netherlands, 1993.

[VR92] A.A. Verrijn-Stuart and G.J. Ramackers. Model Integration of Information Planning Tools. In P. Loucopoulos, editor, *Proceedings of the Fourth International Conference CAiSE'92 on Advanced Information Systems Engineering*, volume 593 of *Lecture Notes in Computer Science*, pages 481–493, Manchester, United Kingdom, May 1992. Springer-Verlag.

[WW93] D.G. Wastell and P. White. Using process technology to support cooperative work: Prospects and design issues. In D. Daiper and C. Sanger, editors, *CSCW in Practice: An Introduction and Case Studies*, pages 105–126. Springer-Verlag, London, United Kingdom, 1993.

[You89] E. Yourdon. *Modern Structured Analysis*. Printice-Hall, Englewood Cliffs, New Jersey, 1989.

[Zav86] P. Zave. Salient Features of an Executable Specification Language and Its Environment. *IEEE Transactions on Software Engineering*, 12(2):312–325, 1986.