

# A Hybrid Interactive Modelling Approach to Flexible Process Support

A thesis submitted to the University of Manchester for the degree of  
Doctor of Philosophy in the Faculty of Humanities

2011

Kevin Mark Finch  
Manchester Business School

# Table of Contents

|   |           |
|---|-----------|
| <b>Table of Contents</b> .....                                      | <b>2</b>  |
| <b>List of Figures</b> .....  | <b>6</b>  |
| <b>List of Tables</b> .....   | <b>7</b>  |
| <b>List of Terms</b> .....  | <b>9</b>  |
| <b>Abstract</b> .....   | <b>10</b> |
| <b>Declaration</b> .....  | <b>11</b> |
| <b>Copyright Statement</b> .....                                    | <b>12</b> |
| <b>Acknowledgements</b> .....                                       | <b>13</b> |
| <b>Chapter 1</b> .....  | <b>14</b> |
| <b>Introduction</b> .....   | <b>14</b> |
| <b>1.1 Problem Area</b> .....                                       | <b>14</b> |
| <b>1.2 Specific Problem Domain</b> .....                            | <b>15</b> |
| <b>1.3 Problem Statement</b> .....                                  | <b>18</b> |
| <b>1.4 Goal, Objectives and Research Questions</b> .....            | <b>18</b> |
| <b>1.5 Research Approach</b> .....                                  | <b>19</b> |
| <b>1.6 Research Contribution and Outcomes</b> .....                 | <b>22</b> |
| 1.6.1 Descriptive Outcomes .....                                    | 23        |
| 1.6.2 Prescriptive Outcomes .....                                   | 24        |
| 1.6.3 Evaluative Outcomes .....                                     | 24        |
| <b>1.7 Organisation of Thesis</b> .....                             | <b>25</b> |
| <b>Chapter 2</b> .....  | <b>26</b> |
| <b>Related Work and Problem Formulation</b> .....                   | <b>26</b> |
| <b>2.1 Process Aware Information Systems Overview</b> .....         | <b>26</b> |
| <b>2.2 Models</b> .....   | <b>27</b> |
| 2.2.1 Model Usage in Traditional PAIS .....                         | 28        |
| <b>2.3 Process-oriented Meta-Models</b> .....                       | <b>30</b> |
| 2.3.1 Process Aspect.....   | 30        |
| 2.3.2 Operational Abstraction .....                                 | 31        |
| 2.3.3 Formality.....  | 33        |
| <b>2.4 Status of Formal Constructs</b> .....                        | <b>33</b> |
| <b>2.5 Model-driven Approaches</b> .....                            | <b>35</b> |
| 2.5.1 Traditional Groupware.....                                    | 36        |
| 2.5.2 Explicit PAIS .....   | 39        |
| 2.5.3 Static Workflow .....   | 40        |
| 2.5.4 Flexible Process Support.....                                 | 42        |
| 2.5.5 Conceptual Approaches for Achieving Process Flexibility ..... | 46        |
| 2.5.6 Flexible Workflow.....  | 48        |

|   |            |
|---|------------|
| 2.5.7 The Interactive Modelling Approach.....                                       | 57         |
| 2.5.8 Integrated Systems.....   | 60         |
| <b>2.6 Web Service Composition .....</b>  | <b>62</b>  |
| <b>2.7 Passive Process Capture.....</b>   | <b>63</b>  |
| 2.7.1 System-Oriented Models for IS Development.....                                | 63         |
| 2.7.2 Process-Oriented Models for Process Modelling.....                            | 65         |
| 2.7.3 Discussion .....  | 66         |
| <b>2.8 Overall Conclusions .....</b>  | <b>66</b>  |
| <b>2.9 Summary .....</b>  | <b>69</b>  |
| <b>Chapter 3 .....</b>  | <b>70</b>  |
| <b>An Integrated Framework for Flexible Process Support.....</b>                    | <b>70</b>  |
| <b>3.1 High Level Requirements.....</b>   | <b>70</b>  |
| <b>3.2 An Integrated Framework .....</b>  | <b>71</b>  |
| <b>3.3 Interactive Modelling Approach to Process Support .....</b>                  | <b>74</b>  |
| 3.3.1 Interactive Models for Process Logic .....                                    | 76         |
| 3.3.2 Interactive Model-Driven PAIS - Conceptual Walkthrough .....                  | 76         |
| <b>3.4 A Hybrid Modelling Language for the Interactive Modelling Approach .....</b> | <b>80</b>  |
| 3.4.1 The Hybrid Modelling Approach .....   | 80         |
| 3.4.2 Declarative Models .....  | 84         |
| 3.4.3 Procedural Models.....  | 91         |
| 3.4.4 Procedural Model Behaviour: Task Execution .....                              | 94         |
| 3.4.5 Computational Supports.....   | 96         |
| 3.4.6 Suitable Conditions for Use .....   | 100        |
| 3.4.7 The Impact of Hierarchical Models on Approach.....                            | 100        |
| <b>3.5 Collaborative Working .....</b>  | <b>102</b> |
| 3.5.1 Techniques to Support Collaboration.....                                      | 102        |
| 3.5.2 Mechanisms to Support Group Working.....                                      | 103        |
| 3.5.3 Collaborative Indirect Work.....  | 104        |
| 3.5.4 Further Anticipated Benefits of Real-Time Collaboration .....                 | 105        |
| <b>3.6 Model Management .....</b>   | <b>106</b> |
| 3.6.1 Life Cycle of Type and Instance Models .....                                  | 106        |
| <b>3.7 Summary .....</b>  | <b>111</b> |
| <b>Chapter 4 .....</b>  | <b>112</b> |
| <b>Implementation: Realisation of Approach in Web-based Environment.....</b>        | <b>112</b> |
| <b>4.1 The Nature of Academic Prototypes .....</b>                                  | <b>112</b> |
| <b>4.2 High Level Requirements.....</b>   | <b>113</b> |
| <b>4.3 Guiding Principals .....</b>   | <b>114</b> |
| <b>4.4 Web-based Synchronous/Asynchronous Groupware .....</b>                       | <b>116</b> |
| 4.4.1 The PowerMeeting Framework .....  | 117        |

|  |            |
|--|------------|
| 4.4.2 PowerMeeting: System Walkthrough .....                       | 119        |
| <b>4.5 The D2P Groupware Application .....</b>                     | <b>122</b> |
| 4.5.1 Model Manager .....  | 123        |
| 4.5.2 Procedural and Declarative Model Editor .....                | 124        |
| 4.5.3 Verification .....   | 131        |
| 4.5.4 Enactment .....  | 133        |
| 4.5.6 Individual and Group Sessions in D2P .....                   | 134        |
| <b>4.6 D2P Application Development .....</b>                       | <b>136</b> |
| <b>4.7 Flexibility through Tool Choice .....</b>                   | <b>137</b> |
| <b>4.8 Approach Summary .....</b>                                  | <b>138</b> |
| <b>4.9 Conclusion .....</b>  | <b>139</b> |
| <b>Chapter 5 .....</b>   | <b>140</b> |
| <b>Evaluation: A Usability Study .....</b>                         | <b>140</b> |
| <b>5.1 The Purpose and Scope of Evaluation .....</b>               | <b>140</b> |
| <b>5.2 Key Metrics .....</b>                                       | <b>141</b> |
| <b>5.3 Data Collection Techniques .....</b>                        | <b>142</b> |
| <b>5.4 Justification of Data Collection Choice .....</b>           | <b>144</b> |
| <b>5.5 Usability Study: Design .....</b>                           | <b>145</b> |
| 5.5.1 Research Context .....                                       | 147        |
| 5.5.2 Hypotheses .....   | 148        |
| 5.5.3 Usability Study Design .....                                 | 150        |
| 5.5.4 Data Analysis and Generating Results .....                   | 156        |
| <b>5.6 Usability Study: Results .....</b>                          | <b>157</b> |
| 5.6.1 Training .....   | 157        |
| 5.6.2 Task .....   | 159        |
| 5.6.3 Task: Task Success (Actual Effectiveness) .....              | 159        |
| 5.6.4 Task: Task Time (Actual Efficiency) .....                    | 161        |
| 5.6.5 Questionnaire (Perceived Effectiveness and Efficiency) ..... | 162        |
| 5.6.6 Other Feedback .....   | 166        |
| <b>5.7 Observations .....</b>                                      | <b>166</b> |
| 5.7.1 Problem Areas .....  | 166        |
| 5.7.2 Chat Transcript Analysis for Group Work .....                | 167        |
| <b>5.8 Data Analysis .....</b>                                     | <b>168</b> |
| <b>5.9 Alternative Usability Study Designs .....</b>               | <b>171</b> |
| <b>5.10 Summary .....</b>  | <b>171</b> |
| <b>Chapter 6 .....</b>   | <b>173</b> |
| <b>Conclusions and Further Work .....</b>                          | <b>173</b> |
| <b>6.1 Contributions of the Study .....</b>                        | <b>173</b> |
| <b>6.2 Limitations and Scope for Further Work .....</b>            | <b>176</b> |

|   |            |
|---|------------|
| 6.2.1 Limitations of the Study.....   | 176        |
| 6.2.2 Scope for Further Work .....  | 178        |
| <b>Appendix A .....</b>   | <b>182</b> |
| <b>Justification of Declarative Expression Technique Choice .....</b>                 | <b>182</b> |
| <b>Appendix B .....</b>   | <b>185</b> |
| <b>Pseudo Code: Verification Algorithms .....</b>                                     | <b>185</b> |
| <b>Appendix C .....</b>   | <b>188</b> |
| <b>Placing D2P Amongst Other Groupware Applications.....</b>                          | <b>188</b> |
| <b>Appendix D .....</b>   | <b>189</b> |
| <b>Procedural Content: Editing Rules .....</b>  | <b>189</b> |
| <b>Appendix E .....</b>   | <b>190</b> |
| <b>The Technology behind PowerMeeting .....</b>                                       | <b>190</b> |
| <b>Appendix F.....</b>  | <b>193</b> |
| <b>Information pack for Participants of Usability Study (Individual Version).....</b> | <b>193</b> |
| <b>Appendix G .....</b>   | <b>196</b> |
| <b>Questionnaire (Group Version).....</b>   | <b>196</b> |
| <b>Appendix H.....</b>  | <b>198</b> |
| <b>D2P Help Documents.....</b>  | <b>198</b> |
| <b>Appendix I .....</b>   | <b>207</b> |
| <b>Screenshots of Models Created during Usability Study .....</b>                     | <b>207</b> |
| <b>Appendix J .....</b>   | <b>210</b> |
| <b>Usability Study Results .....</b>  | <b>210</b> |
| <b>References .....</b>   | <b>212</b> |

Word Count: 62814

## List of Figures

|  |     |
|--|-----|
| Figure 1-1 - Information Systems Research Summary (Hevner, March et al. 2004) .....                                  | 19  |
| Figure 1-2 - Research Process Overview (adapted from Hevner, March et al. 2004 and Vaishnavi and Kuechler 2004)..... | 21  |
| Figure 1-3 - Contribution Development Overview .....   | 23  |
| Figure 2-1 - Generic Procedural Model Example (Sadiq, Governatori et al. 2007).....                                  | 32  |
| Figure 2-2 - Overview of Process Types, Planning Approaches and Supporting Technologies .....                        | 36  |
| Figure 2-3 - Time/Space Matrix for Collaborative Systems.....  | 37  |
| Figure 2-5 - Static Workflow Architecture.....   | 41  |
| Figure 2-6 - Model-driven Approach to Process Support.....   | 43  |
| Figure 3-1 - A High Level View of the Integrated Framework .....   | 72  |
| Figure 3-2 - The Hierarchy of Interactive Models .....   | 75  |
| Figure 3-3 - Pockets of Flexibility .....  | 80  |
| Figure 3-4 - Structure of a Single ‘Pocket’ of Work Specification .....  | 81  |
| Figure 3-5 - Procedural Model with ‘XOR’ Split .....   | 93  |
| Figure 3-6 - Declarative Statement with Procedural Resolutions .....   | 93  |
| Figure 3-7 - Task Execution State Transition Diagram.....  | 95  |
| Figure 3-8 - Example Procedural Model .....  | 96  |
| Figure 3-9 - Constraint State Transition Diagram.....  | 98  |
| Figure 3-10 - Hybrid Models and Hierarchical Structure.....  | 101 |
| Figure 4-1 - Overarching Architecture for PowerMeeting .....   | 118 |
| Figure 4-2 - PowerMeeting GUI.....   | 120 |
| Figure 4-3 - D2P Architecture.....   | 123 |
| Figure 4-4 - Screenshot of D2P and the PowerMeeting Tools .....  | 125 |
| Figure 4-5 - Example Procedural Model in D2P .....   | 128 |
| Figure 4-6 - Example Declarative Model in D2P .....  | 129 |
| Figure 4-7 - Pop-Up Information View for Constraint Nodes .....  | 130 |
| Figure 4-8 - Screenshot of the Verification View .....   | 132 |
| Figure 4-9 - Session Arrangement over Time .....   | 135 |
| Figure 5-1 - Summary of the Usability Study Process .....  | 146 |
| Figure 5-2 - Group and Individual Task Success Results.....  | 161 |
| Figure 5-3 - Individual and Group Task Time Results.....   | 162 |
| Figure 5-4 - Questionnaire results for Individuals .....   | 164 |
| Figure 5-5 - Questionnaire Results for Groups.....   | 166 |
| Figure A-1 - Dimensions of Constraint Specification .....  | 183 |

## List of Tables

|   |     |
|---|-----|
| Table 1-1 – Nature of Work .....  | 16  |
| Table 1-2 – Nature of Process Participants .....                            | 17  |
| Table 1-3 – Nature of Organisation .....                                    | 17  |
| Table 2-1 – A Comparison of Interactive Models and Workflow Approaches..... | 59  |
| Table 2-2 – Existing Problems Overview .....                                | 69  |
| Table 3-1 – Description of Models and their Properties.....                 | 77  |
| Table 3-2 – Summary of Procedural and Declarative Models .....              | 82  |
| Table 3-3 – The Serial Constraint.....                                      | 86  |
| Table 3-4 – The Fork Constraint.....  | 86  |
| Table 3-5 – The Exclude Constraint .....                                    | 87  |
| Table 3-6 – The Include Constraint .....                                    | 88  |
| Table 3-7 – The Precede Constraint.....                                     | 89  |
| Table 3-8 – The Succeed Constraint .....                                    | 90  |
| Table 3-9 – The Resource Constraint.....                                    | 91  |
| Table 3-10 – Constraints Links.....   | 91  |
| Table 3-11 – Procedural Links .....   | 94  |
| Table 3-12 – Verification Summary for Example Procedural models .....       | 99  |
| Table 3-13 – Scope for Group Working.....                                   | 105 |
| Table 3-14 – Models as Governed.....  | 110 |
| Table 3-15 – Models as Governing.....                                       | 110 |
| Table 4-1 – PowerMeeting Terms and Definitions.....                         | 122 |
| Table 4-2 –Procedural Node Summary .....                                    | 127 |
| Table 4-3 – Declarative Node Summary .....                                  | 128 |
| Table 4-4 – Overall Approach Description .....                              | 138 |
| Table 5-1 – Components of an Evaluation Effort .....                        | 141 |
| Table 5-2a – Data Generating Techniques (Close) .....                       | 143 |
| Table 5-2b – Data Generating Techniques (Distant).....                      | 144 |
| Table 5-3 – High Level Summary of Usability Study.....                      | 147 |
| Table 5-4 – Successful Use Matrix.....                                      | 149 |
| Table 5-5 – Potential Outcomes .....  | 149 |
| Table 5-6 – Summary of Data and Metric Collected from that Data .....       | 155 |
| Table 5-7 – Marking Scheme for Models Generated by Participants.....        | 155 |
| Table 5-8 – Scoring System .....  | 156 |
| Table 5-9 – Data Analysis Techniques Summary.....                           | 156 |
| Table 5-10 – Summary of the Training Session .....                          | 157 |
| Table 5-11 – Task Success for Individuals.....                              | 160 |
| Table 5-12 – Task Success for Groups.....                                   | 160 |

|   |     |
|---|-----|
| Table 5-13 – Individual Task Time Results .....       | 161 |
| Table 5-14 – Group Task Time Results .....            | 161 |
| Table 5-15 – Individuals Profile Summary .....        | 163 |
| Table 5-16 – Group Profile Summary.....               | 163 |
| Table 5-17 – Researcher Observations During Task..... | 167 |
| Table A.1 – Simple Constraint Based Problems.....     | 182 |



## List of Terms

|                     |   |
|---------------------|---|
| <b>BPM</b>          | Business Process Management is a field of research and practise addressing all aspects of process-oriented work management.                       |
| <b>EKP</b>          | Emergent Knowledge Process is a type of process characterised mainly by human-orientation and emerging domain requirements.                       |
| <b>PowerMeeting</b> | A Web-based environment providing a central hub and underlying services for various groupware applications.                                       |
| <b>D2P</b>          | Collaborative and graphical model editor for procedural and declarative data structures, delivered as a plug-in for the PowerMeeting environment. |

## Abstract

Process support is a core organisational competence which aims to help people plan and perform their work. This thesis addresses process support for Emergent Knowledge Processes (EKP). EKP can be characterised by human-orientation, emerging domain requirements, compliance requirements and dispersed participants/stakeholders.

Current Process Aware Information Systems (PAIS) do not meet several important support requirements of EKP. Addressing these requirements will create business value. Through a Design Science (Hevner, March et al. 2004) research approach, this thesis addresses that problem. An approach to support EKP is conceptualised and implemented. The main contribution is a prescriptive framework. The framework consists of four components; interactive modelling, a combination of procedural and declarative models, flexible model management and real-time collaborative working. Interactive modelling is an approach to flexible process support where users create, adapt, analyse and enact visual models of their work processes. Underpinning the interactive modelling approach are both procedural and declarative models. Declarative models capture organisational rules. Procedural task-based models capture actual plans of work. Procedural models must comply with the rules established in the declarative models. Flexible model management and real-time collaboration support offer a cooperative working and knowledge management environment for interactive modelling.

The conceptual design ideas have been realised in an academic software prototype. A Web-based groupware application, D2P, was developed as a 'plug-in' for the collaborative, Web-based modelling environment PowerMeeting. It is shown how Web technology can be leveraged for collaborative process support. A contribution is made through further demonstration of the applicability of Web-based solutions for professional, process related business requirements.

As a final contribution, the results of a usability study demonstrate the usability of the D2P and PowerMeeting tools. Empirical evidence in the form of results from a usability study and user questionnaire confirmed that semi-skilled users with a small amount of training can activate the concepts and tools which make up the approach. Of particular importance is the demonstration of the usability of the less familiar declarative modelling paradigm. Scenarios are also provided to demonstrate the applicability of the approach.

## Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning

## Copyright Statement

This thesis is presented under the following copyright guidelines:

- i. The author of this thesis (including any appendices and/or schedules to this thesis) owns certain copyright or related rights in it (the “Copyright”) and s/he has given The University of Manchester certain rights to use such Copyright, including for administrative purposes.
- ii. Copies of this thesis, either in full or in extracts and whether in hard or electronic copy, may be made only in accordance with the Copyright, Designs and Patents Act 1988 (as amended) and regulations issued under it or, where appropriate, in accordance with licensing agreements which the University has from time to time. This page must form part of any such copies made.
- iii. The ownership of certain Copyright, patents, designs, trade marks and other intellectual property (the “Intellectual Property”) and any reproductions of copyright works in the thesis, for example graphs and tables (“Reproductions”), which may be described in this thesis, may not be owned by the author and may be owned by third parties. Such Intellectual Property and Reproductions cannot and must not be made available for use without the prior written permission of the owner(s) of the relevant Intellectual Property and/or Reproductions.
- iv. Further information on the conditions under which disclosure, publication and commercialisation of this thesis, the Copyright and any Intellectual Property and/or Reproductions described in it may take place is available in the University IP Policy (see <http://www.campus.manchester.ac.uk/medialibrary/policies/intellectual-property.pdf>), in any relevant Thesis restriction declarations deposited in the University Library, The University Library’s regulations (see <http://www.manchester.ac.uk/library/aboutus/regulations>) and in The University’s policy on presentation of Theses

## Acknowledgements

My sincere thanks go to Dr. Weigang Wang whose guidance and support made this thesis possible. I can only hope to emulate his professionalism and drive in the years to come. Special thanks also go to Dr. Nadia Papamichail whose valuable feedback and contributions at intervals throughout the research process have been invaluable.

My gratitude is expressed to the support staff at Manchester Business School who helped my navigation through the practical aspects of a doctoral program. Many thanks are given to all who participated within the usability study and the useful feedback given.

A special mention must be given to Jamye for always knowing when it was time to put my pen down.

Finally, my greatest thanks are reserved as always for my parents Mary and Barry, to whom this thesis is dedicated. My appreciation of their unwavering love and support throughout my education goes far beyond what written sentiment can capture.

# Chapter 1

---

## Introduction

This introductory chapter sets the scene for an Information Systems research project. Business Process Management (BPM) forms the overarching problem area. BPM is introduced and the specific problem of supporting Emergent Knowledge Processes (EKP) is identified and summarised. The strategy of how this problem will be addressed is provided. The problem is broken down into three key research objectives, each forming a distinct project phase. The underlying methodology of the research project is Design Science (Hevner, March et al. 2004). A description of the methodology and rationale behind its use is given. The chapter provides a summary of the research contributions which resulted from this research process and closes with a brief description of thesis organisation.

## 1.1 Problem Area

Process-oriented organisations permeate the business environment. For the purposes of this thesis a process is taken to mean ‘a collection of activities which takes one or more kinds of input and creates an output that is value to the customer’ (Hammer and Champy 2001). Process-orientation is the explicit use of process concepts in the planning, performance, monitoring and control of everyday business tasks. Despite the emergence of various agile business forms (e.g. real-time enterprise (Gartner 2003), virtual teams (Powell, Piccoli et al. 2004)), there is still strong reliance on managing work in terms of processes.

Research centred on the business process has a strong tradition in the IS community. A large process-oriented research effort took place in IS during the nineties which focused on process automation through workflow techniques and tools. The emergence of BPM has served to renew this research effort. BPM can be considered to be the next wave of process-oriented research effort (Aalst, Hofstede et al. 2003). BPM is broader in scope than workflow based research. Focus on process automation has decreased with the recognition that a rich array of process types require support from an equally rich array of process-driven IT. Thus, BPM is a broad discipline covering a wide array of techniques and technologies which facilitate one or more of following; the capture, design, enactment,

control, simulation, analysis, improvement and optimisation of operational processes involving humans, organisations, applications, documents and other sources of information (Aalst, Hofstede et al. 2003).

BPM has become a core competency in the current global and highly competitive business environment. The following represent perhaps the key motivations and promises behind the prominence of BPM:

- Organisations use BPM to coordinate the various process aspects (tasks, people and data) and the numerous entities within each aspect.
- Organisations use BPM to ensure that their processes are performed in a consistent and reliable manner.
- Organisations use BPM to facilitate process adaptations in response to changing and emerging business needs.
- Organisations use BPM to ensure compliance with defined organisational and regulatory standards.
- Focus on business processes allows their analysis over time to build up an understanding of that process which can be used to improve and optimise it.

Thus, the promise of BPM is considerable and the reasons for its prominence in research effort clear.

## 1.2 Specific Problem Domain

To begin to carve out the particular problem domain and purpose of this thesis, flexible process support, which is a prominent discipline lying under the umbrella of BPM, is first introduced. Flexible process support essentially seeks to use process-driven IT to help people plan and perform their work. The relevance and applicability of approaches for flexible process support can be maintained by identifying and targeting particular business conditions which will be served.

Tables 1-1, 1-2 and 1-3 develop a problem space covering the users (the nature of people who are doing the work) and the use situation (the nature of the work and the nature of organisations the work belongs to) which are addressed. Characteristics of typical users of process-driven IT and the work they perform are given along with a definition of that

characteristic and an indication of the subset of characteristic values that this thesis addresses.

Table 1-1 – Nature of Work

| <b>Nature of Work</b>    | <b>Possible Values</b>  | <b>Focus</b>                |
|--------------------------|---|-----------------------------|
| <i>Predictability</i>    | Predictable v Emergent<br>How difficult is it to establish a complete process definition far in advance of execution? Emergent processes are highly individual and context dependant (Nutt 1996). Predictable processes can be readily defined far in advance.  | Emergent                    |
| <i>Rigidity</i>          | Static v Fluid<br>Once an initial plan is established to support work, how frequently will it change? Static processes remain the same whilst fluid processes will involve on-the-fly work (re)design.  | Static and fluid            |
| <i>Governance</i>        | Strict v Loose<br>To what extent is the work governed by compliance requirements? Strict processes are employed by organisations striving to meet compliance agendas where business processes are in accordance with a prescribed set of norms (Sadiq, Governatori, et al 2007). Loose processes do not require strong level of governance. | Strict and loose            |
| <i>Complexity</i>        | Low v High<br>How many entities (e.g. tasks and resources) are involved in process definition? Highly complex processes coordinate many interrelated entities and dictate the necessity of some notion of process-orientation. Low complexity processes have smaller numbers of entities to organise.                                       | Medium                      |
| <i>Grain</i>             | Strategic v Tactical v Operational<br>At what grain is process definition required? At the strategy level, high level goals are articulated. These are broken down until the operational plans (finer grain tasks) which can realise the goals are generated (Mason and Mitroff 1973).  | Operational                 |
| <i>Process Knowledge</i> | Explicit v Tacit<br>How far can process knowledge be encoded i.e. captured in concrete plans? Explicit process knowledge is readily captured and articulated. Tacit knowledge resides in the mind of human actors and is not readily articulated (Courtney 2001).   | Explicit                    |
|                          | Distributed v Centralised<br>To what extent is process knowledge distributed? Process knowledge can belong to a single actor or shared between multiple actors.   | Distributed and centralised |
| <i>Repetitiveness</i>    | Unique v Recurring<br>Will similar work be repeated frequently? Unique work is usually called a project whilst recurring work is called a process.  | Unique and recurring        |
| <i>Frequency</i>         | Low v High<br>How many pieces (unique or recurring) of work will be performed? Low is taken to mean typically several per day. High is taken to mean hundreds a day.  | Low                         |



Table 1-2 – Nature of Process Participants

| <b>Process Participants</b> | <b>Possible Values</b>   | <b>Focus</b>                 |
|-----------------------------|--|------------------------------|
| <i>Actors</i>               | Human v Technical<br>Who will perform the work? Human workers usually perform tasks which are knowledge intensive and rely on the skill/experience of the person. Technical resources can be split into software and hardware which perform tasks automatically (Muehlen 2004).                        | Human                        |
|                             | Novice v Semiskilled v Skilled<br>To what extent are human workers skilled and experienced in the use of process-driven IT systems? Novice users have little or no skill or experience in process-oriented concepts or tools while skilled users are competent in process-oriented concepts and tools. | Unpredictable knowledge base |
| <i>Team Dynamics</i>        | Individual v Group<br>How do process participants perform process-related work? Work can be performed individually or in groups. Group working is also referred to as collaborative, coordinated or collective work (Powell, Piccoli et al. 2004).   | Individually and groups      |
|                             | Distributed v Co-located<br>To what extent are team members geographically divided? Dispersed teams are often referred to as virtual teams. Co-located workers work in close physical proximity.   | Distributed and co-located   |

Table 1-3 – Nature of Organisation

| <b>Organisation</b> | <b>Possible Values</b>   | <b>Focus</b>                   |
|---------------------|--|--------------------------------|
| <i>Scope</i>        | Inter-organisational v Intra-organisational<br>Is the process performed by multiple cooperating enterprise partners? Inter-organisational processes usually involve team members from different partner organisations working collaboratively on a joint project (Wang, Finch et al 2009). Such forms are particularly employed by small and medium sized organisations that pool resources to provide a coherent service to compete with larger organisations. Work is being increasingly performed by distributed networked organisations, also known as collaborative commerce, partnerships and networks. Intra-organisational processes are fully enacted within the bounds of a single organisation. | Inter and intra-organisational |
| <i>Term</i>         | Temporary v Permanent<br>To what degree is the organisational form long lasting? Temporary organisations are created for the performance of processes over a short space of time and then disbanded. There is still a great need for temporary integration across organisational boundaries (Collins, Ketter et al. 2010). Permanent organisations enact processes over a long term.   | Temporary and permanent        |

The nature of the processes targeted by this thesis has been developed by Tables 1-1, 1-2 and 1-3. For the remainder of the thesis, processes having such properties will be referred to as Emergent Knowledge Processes (EKP) (Markus, Majchrzak et al. 2002).

Processes with these properties can be found in diverse application domains making research which addresses their needs widely applicable. Sectors include IT services, research and development, health care, operations management, customer relationship management, public sector (government agencies), academia and crisis management. Concrete examples include administrative procedures (such as purchasing and recruitment), call-centre support and medical processes (patient treatment programs). The ideas presented in this thesis are equally applicable to project work commonly found in the IT, consultancy and building industries for example.

### 1.3 Problem Statement

The promise of BPM, as described in Section 1.1, has not been fully realised. Model-driven IT systems are well established for facilitating certain procedures. One remaining challenge is to develop a model-driven approach (techniques and tools) which can be applied to support EKP (Markus, Majchrzak et al. 2002). How can groups of people be better supported in the planning, performance and management of EKP?

Simply summarised, the research effort seeks to gain an understanding of a use situation and develop an approach to better serve it. The provision of flexible support which can meet the requirements of EKP is an open, current and challenging problem. The characteristics established by Tables 1-1, 1-2 and 1-3 are context-free which maintains a wider applicability of research outcomes.

### 1.4 Goal, Objectives and Research Questions

The ultimate goal of this research is the conceptualisation, realisation and evaluation of an IT based process support solution which addresses the needs of EKP. This goal can be broken down into several objectives and research questions.

The first objective of the work is a critical analysis of the ability of current techniques and technologies to support the planning and enactment of EKP.

- What critical issues and limiting factors can be identified in the current crop of PAIS in terms of their application to EKP?

The second objective of the work is to develop techniques and tools to support users in the planning, performance and management of EKP with specific regard to any identified critical factors.

- How should models be deployed and managed to drive day-to-day work and achieve a balance between process control and flexibility? Furthermore, how is this achieved whilst maintaining usability of the PAIS?
- How can group working be better incorporated into PAIS?
- What IT environment can be rapidly developed, extended and deployed for developers as well as affording rapid connection and consumption for users?

The third objective of the work is the evaluation of any proposed solution. Evaluation efforts will consider the performance of designed artefacts in their ability to meet predefined criteria. Criteria in this project can be broadly summarised as usefulness and usability.

- Is the approach useful to practitioners who regularly execute EKP?
- Is the approach usable by novice and semi-skilled knowledge workers?

## 1.5 Research Approach

Information Systems is an interdisciplinary subject encompassing business, people and technology (Mason and Mitroff 1973). A broad overview of the nature of research in IS is given in Figure 1-1, taken from (Hevner, March et al. 2004).

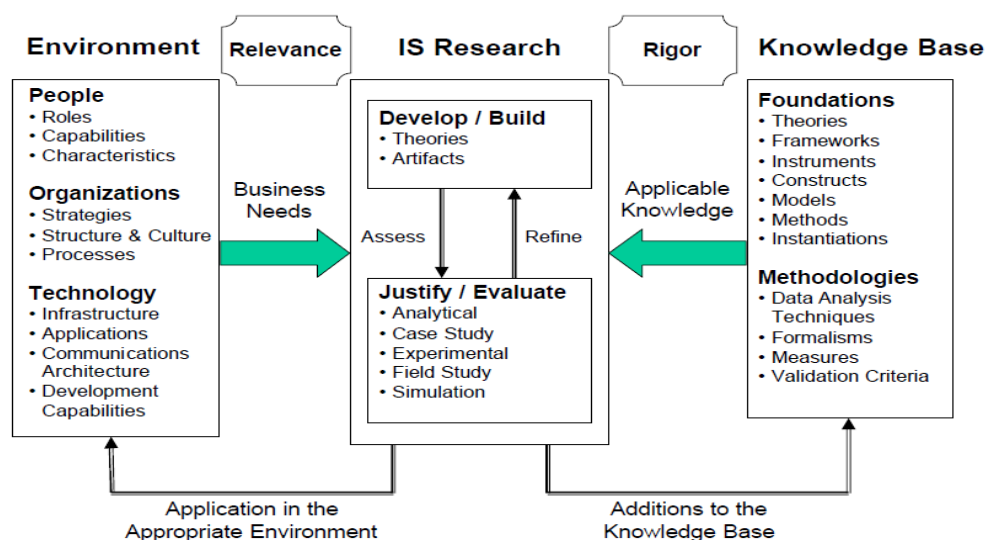


Figure 1-1 - Information Systems Research Summary (Hevner, March et al. 2004)

This research project addresses a ‘wicked’ problem. Wicked problems are characterised by the impossibility of a definitive formulation due to unstable requirements, ill-defined environmental contexts and a myriad of complex and interacting psychological and social factors (Courtney 2001).

To address such problems, it has been suggested that the move forward and development of understanding is achieved incrementally through ‘*satisficing*’ solutions (H.A.Simon 1996). This is the design and evaluation of artefacts where contribution is made gradually through demonstration of utility.

A constructive research approach excels in this situation as it is primarily a problem solving research paradigm. “The constructive research approach is a research procedure for producing innovative constructions, intended to solve problems faced in the real world and, by that means, to make a contribution to the theory of the discipline in which it is applied” (Caplinskas and Vasilecas 2004).

A thorough formulation of a constructive research approach is provided by Hevner’s Design Science (Hevner, March et al. 2004). Knowledge and understanding is developed through the building and evaluating IT artefacts including constructs, models, methods and instantiations designed to meet the identified business need. Contribution is then assessed by the artefacts ability and utility in meeting the identified need.

This thesis contributes to information systems research via the method of Design Science (Hevner, March et al. 2004). Figure 1-2 is based on the framework of Figure 1-1 to give an overview of how this research effort applied the Design Science (Hevner, March et al. 2004) research paradigm.

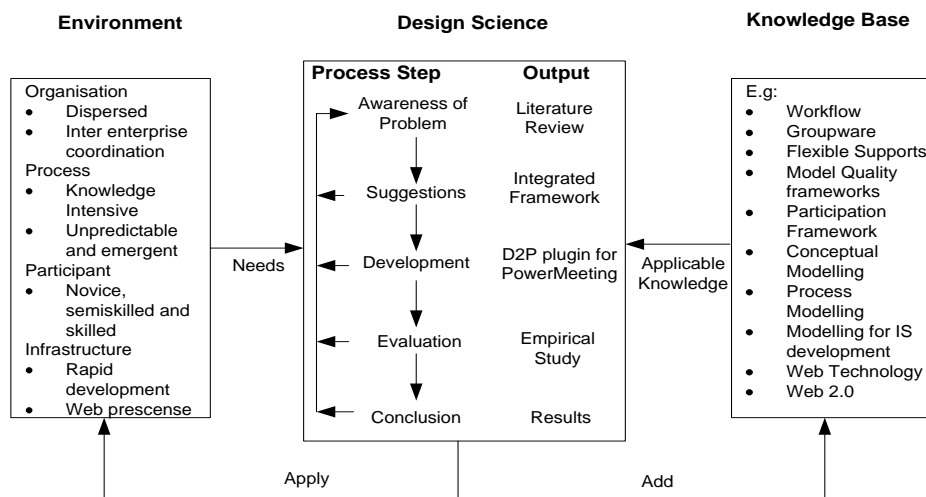


Figure 1-2 - Research Process Overview (adapted from Hevner, March et al. 2004 and Vaishnavi and Kuechler 2004)

It is important to note that most Design Science (Hevner, March et al. 2004) research projects do not follow a single cycle of a strict sequence of process steps as the idealised commentary provided by this thesis may indicate. There is often interplay between the process steps. For example, the exploration of a tentative suggestion may help redefine the problem or evaluation results may feed back into design. Each step contributes to the development of understanding, taking insight of problem and solution from fuzzy into focus.

To give an example in terms of this research project, one suggestion was the use of a structured process annotation technique using Object Constraint Language. After some initial development this line of investigation was discontinued due to the strong expertise required to use the approach which cannot be assumed in the problem domain. A second suggestion was the use of XCHIPs as a test bed for the approach. Again this strategy was abandoned as its lack of Web presence was not in alignment with the needs of EKP. Such iterations led to a greater understanding of the problem situation and informed subsequent developments.

Design Science (Hevner, March et al. 2004) is a relatively modern research method which does not fall in either of the traditional categories of positivist or interpretive research. It stands on its own as an alternative method. However, Design Science (Hevner, March et al. 2004) can and increasingly does borrow techniques from these categories of research (Detmar, Gefen et al. 2004). For example, design research can find acceptable validation of

a new design through a proof of concept. In many cases this validation can be strengthened by closer empirical scrutiny of artefacts using positivist techniques such as controlled experimentation, field work, case studies and surveys.

## 1.6 Research Contribution and Outcomes

A flexible, model-driven solution is applied to the support of EKP. This thesis contributes to a larger research effort which systematically examines model-driven techniques and technologies in various problem domains. The contributions made by this thesis have relevance for both practitioners and other researchers:

- *Practitioners* - use research outcomes to determine if application of the work can help improve current work practises (i.e. consider potential to adopt and deploy research findings).
- *Researchers* - use research outcomes to influence future research by extracting opportunities for development. Researchers could be both native (those directly associated with the development and evaluation of the research) and foreign (those external from it, but interested in research results such as other academics looking to build upon research and students looking to learn from research).

The contribution of this thesis is the demonstration that the interactive modelling approach to process support can be extended to address the needs of EKP with a hybrid interactive modelling language and Web technology whilst maintaining usability.

Several research outcomes substantiate this claim of contribution. A summary of the outcomes made by this thesis are organised around the objectives given in Section 1.4. The following discussion includes both descriptive and prescriptive outcomes. A small portion of this work is description-driven research (novel accounts of existing phenomena in the subject domain). This content contributes by illuminating the problem domain in a new way to better understand the nature of problem. The larger portion of this work is prescription-driven research which provides solutions and recommendations for the described problem. Recommendations come as a pair of problem situation and approach (a technique or tool for example).

Figure 1-3 below gives an overview of the research process in terms of several artefacts which were generated as a result of it.

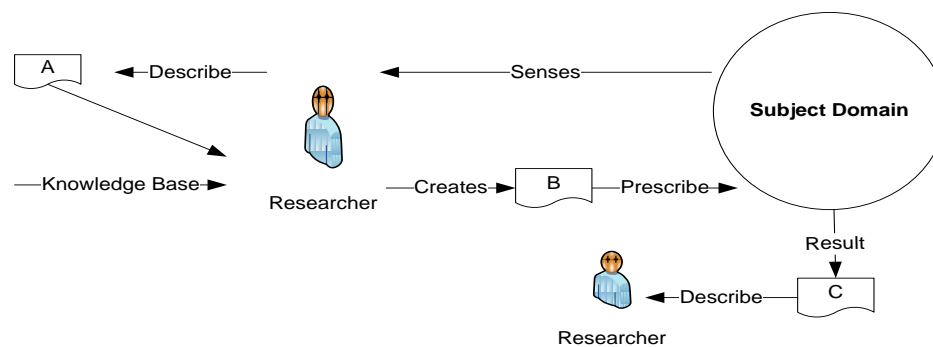


Figure 1-3 - Contribution Development Overview

The following sections summarise what each artefact (A, B and C) is and discuss how far objectives 1, 2 and 3 were met by those artefacts.

### 1.6.1 Descriptive Outcomes

Artefact A is the result of Objective 1. Artefact A consists of a description of EKP and the organisational forms which typically use them. This description was used to analyse the ability of the current range of process-driven IT solutions to meet the support needs of EKP. Several inter-related deficits limit applicability to EKP:

- *The balance between process flexibility and process control.* Providing a workable balance between flexibility and control is an enduring organisational challenge and one which current PAIS have not effectively mastered.
- *The potential for collaborative working in process-related tasks.* Real-time collaboration support is required for joint process-related activities (e.g. the creation, analysis and modification of plans) (Wang, Finch et al. 2009). Some existing process-driven IT integrate communication tools but lack rich collaborative working opportunities.
- *The need for integrated systems which can support a diverse array of processes.* It is becoming clear that most organisations need integrated support for various kinds of processes (Bernstein 2000). This should be addressed without increasing the complexity of the working environment (e.g. the number of tools on the user's desktop). Although a range of support can be provided, current PAIS fail to do this effectively in a single integrated system.
- *Ability of users to rapidly connect to process work.* IT support should facilitate the rapid connection of involved parties to the process environment. Dominant desktop

applications can be cumbersome to deploy (for administrators) and consume (for users) which discourages take up and slows down inter-organisational connections.

### 1.6.2 Prescriptive Outcomes

Artefact B is the result of Objective 2. It is the researcher developed solution to the problem situation identified by artefact A. Contribution to BPM is made through the conceptualisation and implementation of an integrated framework which can support EKP:

- *Conceptualisation* - It is shown how the interactive modelling approach can be widened for applicability to EKP. This involves an innovative hybrid modelling language to underpin modelling activity using interactive graphical models of work, provision of strong collaborative opportunities amongst team members in model interaction and a flexible model management approach which addresses the impact of a hybrid modelling language and collaborative working.
- *Implementation* - A software prototype called D2P implements the conceptualisation. With D2P a contribution is made to Web development by showing how Web technology can be leveraged for collaborative process support. In particular, it is described how full-fledged interactive real-time process-aware groupware applications with desktop like responsiveness and user experience can be implemented on the Web.

### 1.6.3 Evaluative Outcomes

Artefact C is the result of Objective 3. It is the results obtained from the application of artefact B to the subject domain:

- The usability of the D2P tool is demonstrated empirically through a usability study. The study showed how users can interpret declarative graphical models to generate a compliant procedural model both working individually and collaboratively. This is particularly important as of the current lack of empirical data on usage of declarative modelling concepts. Furthermore, this demonstrates the potential of formal usability studies for the assessment of process-driven IT, which thus far has been an underused evaluation technique in the field.
- The usefulness of the approach is demonstrated with an ongoing detailed scenario.



## 1.7 Organisation of Thesis

The remainder of the thesis is organised as follows:

Chapter 2 presents a comprehensive review of work related to the problem domain of flexible process support. Basic concepts are defined and described to give context to a thorough description of process-driven IT systems and their suitability for application to EKP. Related work also draws on concepts from fields such as process modelling and modelling for IS development. The review is rounded up with the identification of critical limitations in the ability of current PAIS to provide flexible process support for EKP.

Chapter 3 describes a conceptual framework which establishes a novel approach to flexible process support. The interactive modelling approach to process support (Jørgensen 2004) is tailored in several ways. First it is underpinned with hybrid process modelling language harnessing both procedural and declarative modelling concepts. Second, a strategy for collaborative working amongst process participants is developed. The description shows how ideas from group working can be seamlessly integrated in to PAIS. Finally, a flexible model management is presented which pays careful consideration to the impact of a hybrid modelling language and a collaborative working strategy on model management.

Chapter 4 details a Web-based implementation. A process-aware groupware application called D2P acts as a plug-in for the PowerMeeting system (Wang 2008). Together, these research prototypes realise the way of working described in Chapter 3 in a functioning software system.

The bulk of Chapter 5 describes the design and execution of an empirical usability study focusing on the usability of D2P. Initial results are reported which show the great the potential of hybrid languages, collaborative working and Web-based delivery for PAIS.

Chapter 6 reviews the research contributions made to BPM and IS. An important part of this chapter is the identification of directions for future related research.

# Chapter 2

---

## Related Work and Problem Formulation

Research builds upon and is influenced by previous research. For the purposes of this thesis it is necessary to first establish basic process related concepts and terminology. These basic concepts are referred to in the critical examination of current techniques and technologies and their ability to provide process support for EKP. A broad view of related work was required. The survey covers groupware, static workflow, flexible workflow, interactive modelling approaches and integrated systems. Furthermore, relevant work in conceptual modelling (in the process modelling and IS development fields) are briefly covered.

The outcome of this chapter is the identification of several critical areas which severely hamper the ability of current techniques and technologies to provide flexible support for EKP.

### 2.1 Process Aware Information Systems Overview

The separation of application logic and application data is a common pattern in IS. Application logic is usually an underlying, reusable application framework established by professional developers. For example, a desktop publishing program such as Microsoft Word forms application logic and the written documents form application data. In other words, environments are created for other people to design solutions (Fischer, Giaccardi et al. 2004).

Process Aware Information Systems (PAIS) (Dumas, Aalst et al. 2005) adhere to this pattern; generic applications which separate application logic from application data (business process logic) (Weber, Sadiq et al. 2009). Process logic is used to coordinate work seeking to ensure a synchronised, harmonious, organised group activity. Domain and modelling experts use the application logic to define the business process logic often through some graphical modelling technique.

This separation of two distinct kinds of logic contributes to the flexibility of PAIS as the process logic can be changed independently of the application logic. This is not the case for

traditional process-aware information systems which mix application and business process logic at a low level (i.e. process logic built into code).

PAIS can be generally described as model-driven software systems supporting some aspect of the capture, analysis, enactment and management of business processes (Dumas, Aalst et al. 2005). Three simple elements of a PAIS can be identified:

- A model captures process-related information (e.g. scheduling of tasks and assignment of resources that perform them).
- The model is used in some way to guide process performance. There are several deployment techniques (such as workflow and interactive models) which can be used to drive work performance.
- A software tool implements the above services.

## 2.2 Models

The description of PAIS given above demonstrates their model-driven behaviour. Let us establish what the term model means.

A model is an abstract, often graphical, representation of some target phenomenon (Wand and Weber 2002). The target phenomenon refers to some aspect of the past, present or future physical and social world around us. It is an abstract representation as models describe only particular characteristics of the target phenomenon. We can say that models are a filtered version of reality to help model users study certain important features of that reality (Wang 2008).

Models come in many variants, from physical artefacts (e.g. made of Plasticine) to conceptual artefacts such as two and three dimensional graphical diagrams (e.g. on paper or a virtual work surface). The model can try to capture the nature of an existing or existed phenomena (as-is or as-was) or try to influence the form of phenomena which is yet to exist or happen (to-be). Model usage promotes early detection and correction of errors for all models uses (Wand and Weber 2002).

Whatever the purpose of the models themselves, modelling has two main goals. First, the faithful representation of target phenomena (alignment between the target phenomena and the model). Second, the faithful interpretation of the model (alignment between the model

and the understanding afforded by that model) (Wand and Weber 2002). These basic goals shall be revisited throughout the remainder of the thesis.

The use of models is ubiquitous. Typical examples include the blueprints for a new house, a circuit diagram and the scale model of a car. These examples focus on the representation of a physical artefact to reason about the form of that physical artefact; a house, a circuit board and a car. Models can be used to reason about the form of less tangible phenomena. In the domain of IS, the highest ranked purposes for which modelling was undertaken was database design and management (Davies, Green et al. 2006). However, it was also shown how model use in IS is diversifying, particularly with regards to BPM. A recent and renewed urgency of research into model and modelling related activities in the BPM domain has sought to address the challenges of the complex modern enterprise environment (Aalst, Hofstede et al. 2003).

### 2.2.1 Model Usage in Traditional PAIS

Taxonomy of three different types of work (A, B and C) for knowledge management has been captured in (Engelbart 1992). The taxonomy is applicable to BPM as a specialised form of knowledge management. To help gain an understanding of BPM, the various roles that models play in BPM solutions can be explored by relating them to each type of work from Engelbart's framework.

The following discussion is in terms of the operational goals of a BPM effort (such as the hiring of a new employee, the purchase of a car, the rehabilitation of a patient etc). The examples given will be in terms of workflow systems (discussed further in Section 2.5.3). Traditionally, IT professionals generate a PAIS (C level work) for professional users to plan (B level work) the work of performers (A level work).

#### **C Level: Indirect Work (Application Logic)**

Models are used at this level to help reason about and develop the application logic of the PAIS. Typical models include system analysis and design models and meta-models which are then eventually realised in low level code. Model use here is passive as the model is essentially created and then exists outside of its subject; domain phenomena might change but the model, in general, does not react to this change (Greenwood, Robertson et al. 1995).

This is usually the realm of the IT professional. For example, IT professionals may undertake a IS development effort to create a workflow system. The majority of this chapter represents coverage of other C level efforts.

### **B Level: Indirect Work (Process Logic)**

Models are used at this level within a PAIS as a device to establish process logic; plan, reason about and manage the coordination of actual work which will eventually be performed through the creation, analysis and adaptation of process models. This is usually the realm of process and modelling experts. For example, the workflow system (and its meta-model) created by the C level effort is used to create a reusable purchasing process or a hiring process. This chapter describes the major approaches to process coordination.

An explicit coordination effort is necessary due to complex business endeavours; to ensure process goals are being met and individuals know what they are expected to do and what is expected of others. For smaller processes, the problem of coordination is not as acute as activities can be articulated largely or entirely by means of direct interaction.

### **A Level: Direct Work (Process Logic: Work Performance)**

The process logic captured in models created by B level efforts are used at this level to drive and guide actual performance of work (the model is executed to dictate what action should/will occur). The actual work may be performed by social (e.g. a physical task such as filling out a form) or technical actors (e.g. the consumption of a Web Service). The model is integrated into an information system and can be enacted in different ways (Krogstie 2007):

- a) *Manually* - people interpret and execute the process model and use it as a source of instruction as to what work should be performed next. The PAIS offers no active support or enforcement.
- b) *Automatically* - a software component interprets and executes the model; the system plays an active role in enforcing a 'script'.
- c) *Interactively* - the computer and the human users co-operate in interpreting and executing the model.

This is traditionally the realm of work performers. For example, the model created in the B level effort is deployed to manage the purchase of an actual company car.

## 2.3 Process-oriented Meta-Models

Meta-models give modellers a set of tools or re-usable schemes for creating models which adhere to a specific modelling technique. Meta-models are made up of concepts specifically conceived for a particular domain to form a modelling technique. Models which subscribe to a meta-model use the concepts and concepts linking rules it establishes.

Typical concepts in process-oriented meta-models include tasks and actors, precedence links and assignment links. Concept linking rules restrict how they may be associated e.g. a 'precede' relationship may only exist between two tasks. An individual process model then instantiates these concepts as per domain requirements; an arrangement of tasks, precedence links, resources and assignment links define the actual work to be performed and the resources that will perform it.

Meta-models afford precision and consistency in articulation and interpretation of models as well as scope for automated computational supports on the model (Wang 2008). We next use meta-model features (the aspect, the operational abstraction and the level of formality used) to help describe process modelling capability.

### 2.3.1 Process Aspect

As described, each model is an abstraction, only characteristics of the target domain that are essential to the problem being studied are modelled. Which characteristics of the process should appear in the model and which should be ignored? Different meta-models have different answers to this question. Process related knowledge is usually based around one or more the following three aspects:

- *The Task Aspect* - This perspective facilitates definition of the tasks and the order in which they can be executed (Aalst, Hofstede et al. 2003). In other words, asserting what happens and when it will happen.
- *The Resource Aspect* - This perspective facilitates definition of which resources can and will perform which tasks. In other words, who may do what (authorisation models) and who will do what (actual allocations)). A resource may be a person, typically someone filling an organisational position, such as a manager or a supervisor, or it may be another form of resource such as a database or computer application (Russell, Hofstede et al. 2005).

- *The Data Aspect* - The data perspective facilitates definition of the operational data which is needed as input to and provided as output from tasks. Control flow is often used to pass data from one task to another. However, some data may be passed through tasks that may not need it. Decision data is a subset of operational data, which is needed by choice coordinators to make control flow decisions as to which tasks will be performed when there exists choice (Russell, Hofstede et al. 2004).

An enterprise has to be viewed from several perspectives if it is to be fully described and understood. Traditionally knowledge management has been addressed through the use of multiple conceptual models, each emphasising different aspects of the subject domain. This may not allow us to have a ‘big-picture’ of a system (Savolainen, Beeckmann et al. 1995). To understand something in isolation is not to truly understand as relationships and interactions are not presented.

This motivates the combination of the above aspects in a single meta-model (Lillehagen, Wang et al. 2004). These are called holistic meta-models. Holistic models make it easier to visualise and maintain consistency of design (Wang 2003). Often holistic meta-model are task centred, the task aspect acting as a hub drawing in relevant connected information.

### 2.3.2 Operational Abstraction

Meta-models can be described in terms of the modelling paradigm they use. A useful dichotomy is the procedural and declarative modelling camps (whilst it is recognised that separation between declarative and procedural modelling languages is not always clear cut, it remains a useful device to facilitate discussion.)

#### **Procedural**

Procedural modelling techniques (also known as imperative (Pesic and Aalst 2006) and transformational (Jørgensen 2004)) dominate process modelling. Heavily prescriptive control flow graphs are used to assert a particular flow of work. Places (which can represent tasks) and transitions are basic the ingredients of the control flow graph. Each task takes inputs, which it then transforms to outputs. Input and output relations thus define the sequence of work. Typical examples of this style of process capture include Data Flow

Diagrams (Hamlet and Maybee 2001), Activity Diagrams (Hamlet and Maybee 2001) Event-Driven Process Chains (Scheer 2000) and Petri nets (Aalst 1998).

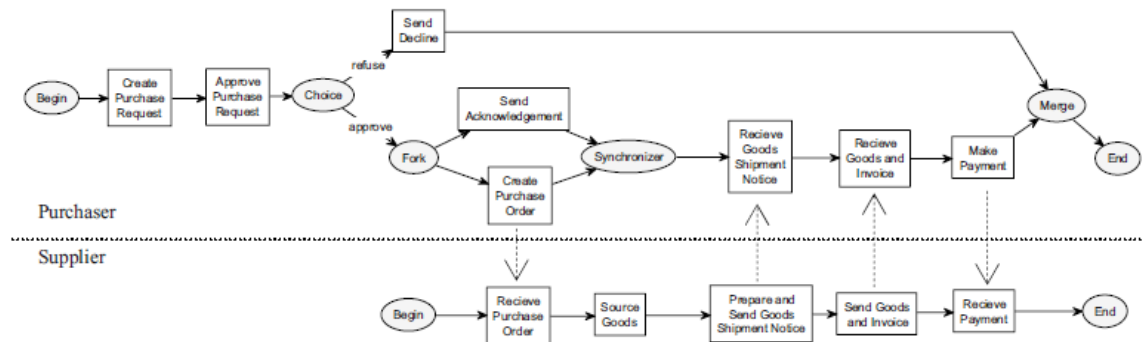


Figure 2-1 - Generic Procedural Model Example (Sadiq, Governatori et al. 2007)

A typical procedural model is shown above. Languages usually implement some branching (choice and fork) and join (merge and synchronise) logic. See (Aalst, Hofstede et al. 2003) for a full description of branch and join logic.

Procedural models lend themselves to goal-directed decomposition. It is important to start with a business process model which fully articulates the vision and goals for the new process (Kueng and Kawalek 1997). A goal is some desired end state in the process domain. These goals may begin as fuzzy and vague ideas. Goal-directed decomposition allows larger, vaguer goals to be broken down until operational detail is achieved.

Graphs are then made up of complex (content in the form of a sub-process) and atomic (no sub content) tasks. For example, from Figure 2-1, the task 'Make Payment' could be made up of a sub-tasks including, 'Choose Payment Method', 'Authorise Payment' and 'Pay'. Extended Information Control Nets (Nutt 1996) and goal-based process models (Kueng and Kawalek 1997) are examples of this way of working.

### Declarative

Declarative languages (also known as constraint and rule based languages) are more descriptive than prescriptive. Intentions are articulated in varying level of detail (strategy, goals, rules, requirements and constraints) which do not prescribe a particular plan of work, but establish what must be achieved and adhered to by work. The precise and actual details of the work plan are placed under performer control. Declarative information can be articulated in many ways. Examples include free language such as English, controlled



languages such as OCL (Hamlet and Maybee 2001), visual languages (employing graphical and image notations), first-order logic language and hybrid languages (some combination of visual and natural languages). For concrete examples see (Dourish, Holmes et al. 1996; Glance, Pagani et al. 1996; Aalst and Pesic 2006).

### **Hybrid**

As seen in the aspect description, an enterprise has to be viewed from several perspectives if it is to be fully described and understood. Recent approaches have started to combine the procedural and declarative paradigms of modelling. There is a vivid interest among practitioners and commercial software vendors in the confluence of business rules and business process modelling (Debevoise 2005). Concepts have been presented investigating how declarative information such as risk assessment (Muehlen and Rosemann 2005) or compliance data (Sadiq, Governatori et al. 2007) can be specified as part of a procedural model.

### **2.3.3 Formality**

Meta-models can vary with the amount of structure they prescribe. Structured content is used for strong coordination where the process is formally defined e.g. tasks connected with precedence links and resource allocations. Technical process analysis and enactment can be applied to structured content. Semi-structured content is used for looser coordination. The model may use concepts such as tasks but are arranged into less formal manner e.g. collections, clusters or lists (Jørgensen 2004). Unstructured content is an informal artefact based on for example ideas, annotations, drawings and conversations.

Frequently meta-models will afford variety in the degree of structure required in process articulation. One level of formality will often not support articulation requirements, for example, if unstructured only then the purpose of process orientation is lost and the scheme purely serves as free modelling (e.g. on a whiteboard). Alternatively, frequently the insight into a process may not be adequate to make a full, formal specification.

## **2.4 Status of Formal Constructs**

The presumption that pre-defined organisational constructs (such as formal plans and procedures) determine action has been subjected to critical examination (Suchman 1995).

The role of such constructs in cooperative work is still far from understood. What is clear is that there is evidence of formal constructs being useful in differing ways.

### **As Maps**

Strong concern has emerged about the credibility of formal organisational constructs when used in a wholly prescriptive capacity. This is based in the perspective that process models are impoverished idealisations unable to encode the richness of the process work. It has been suggested the limitations and opportunities presented by the current environment is what drives work, rather than strict plans. Plans are thus "*resources for situated action, but do not in any strong sense determine its course*" (Suchman 1987).

### **As Scripts**

Other studies result in a different perspective as to how formal constructs are used by actors in everyday work activities. The introduction of formal constructs for coordination purposes is generally in response to the problem posed by the management of complex cooperative work. Scripts are a pre-computation of interdependencies among activities which are usually enact-able in some way. They are positioned as strong determinants of action as they are highly prescriptive (offering valid process actions whilst excluding disallowed actions).

Evidence has been presented that suggests computer systems can be successfully designed to support cooperative work of the knowledge worker by providing representations of formal organisational constructs (Schmidt 1999). Under conditions of limited resources, practical exigencies, and social accountability they rely on the stipulations of the script, when one is at hand, in order to get the job done, unless they have good reasons not to do so (Schmidt 1999).

### **Discussion**

We can conclude that organisational constructs such as plans always play some role in driving work. Direct work is often script-driven. When that script is flexible, script-driven work can be maintained when work requirements change. Even where actual activities differ from the work plan, in the formulation of that activity, the script still makes a contribution as one source of information which has been used to determine the course of action.

## 2.5 Model-driven Approaches

Let us turn towards concrete approaches for how models can be deployed for work planning and performance in PAIS. A diverse array of process-oriented techniques and technologies has been developed over several decades. We discuss the relevant approaches with respect to the broad requirements of EKP as identified in the introductory chapter.

Specificity is used to classify PAIS. Specificity refers to the amount of structure which is bestowed upon the user in the performance of the process, in other words the level of autonomy afforded by the approach (Bernstein 2000). At the extremes of the spectrum we have two basic perspectives for coordination support:

- *Low Process Specificity* - Leans towards minimal prescription of work as the technology helps people freely execute work in a manner dictated by the situation which is often intuitive, less structured and rapidly opportunistic. Here workgroup values are subscribed to; helping people complete work in a locally-determined fashion (Bernstein 2000), (Suchman 1987).
- *High Process Specificity* - Leans towards a fixed prescription of work as structured predefined processes emphasise on standardisation, speed and efficiency. Here workflow values are subscribed to; work is completed in a centrally-determined fashion (Schmidt 1999).

As shown in Figure 2-2, a deeper insight into the problem space can be gained by using specificity as baseline to help draw in other descriptors drawn from (Nutt 1996) and (Jørgensen 2004).

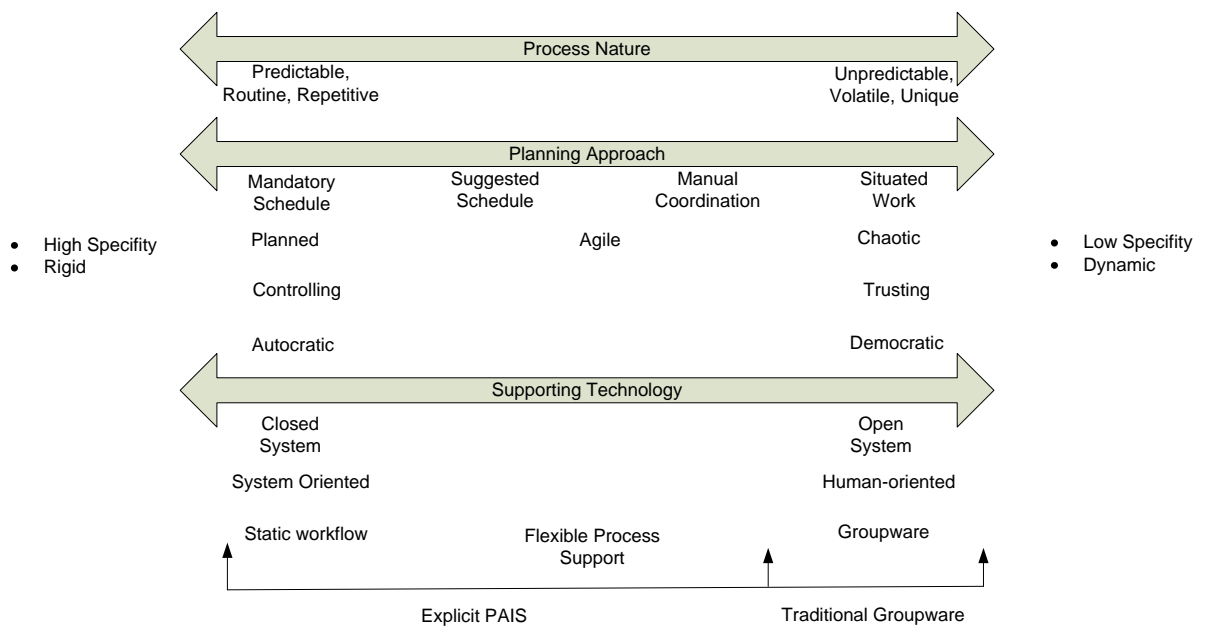


Figure 2-2 - Overview of Process Types, Planning Approaches and Supporting Technologies

Concrete examples of EKP that exist in the low to mid area of specificity include patient treatment plan, crisis management and response. It is interesting to note that Figure 2-2 is an idealised view of the problem space. Frequently in practice, processes are paired with tools and planning approaches whose characteristics are not compatible.

The spectrum of supporting technology from Figure 2-2 is now examined with emphasis given to the various techniques for flexible process support. Full coverage of the space is given; however, focus is on mid-high end of each spectrum.

### 2.5.1 Traditional Groupware

Groupware are software systems that use computers and networks to assist groups of workers in coordinating and performing their work (Dix, Finley et al. 2004). These systems intentionally lack explicit process representations and thus do not strongly coordinate the work process participants perform (Wang, Finch et al. 2009).

The ultimate goal of groupware, from a business perspective, is the same as other technologies - to help people get work done. A more rudimentary coordination mechanism is provided for this end; the management of the many individual interactions of process

participants with other process participants and process participants with artefacts of work which together make up both the direct and indirect work effort.

Different kinds of groupware have been targeted at different kinds of tasks. The traditional framework for the classification of groupware comes in the form of the time-space matrix below (Dix, Finley et al. 2004).

|       |         | Time |           |
|-------|---------|------|-----------|
|       |         | Same | Different |
| Space | Close   | A    | B         |
|       | Distant | D    | C         |

Figure 2-3 - Time/Space Matrix for Collaborative Systems

Zones A and D are the realm of synchronous groupware. Zones B and C are the realms of asynchronous groupware.

Groupware can be classified by function. The two main functions are communication and collaboration. Whether communication or collaboration-oriented, each system could potentially occupy any zone presented in Figure 2-3.

Communication refers to the basic ability to exchange information. This usually involves some message passing facility (e.g. text, sound and pictures). Web-based systems include various Webmail facilities, Facebook (Facebook 2010) and Wiki (Leuf and Cunningham 2001) which support informal communication asynchronously. Instant Messaging tools (e.g. (Microsoft 2010)) and A/V conference tools (e.g. (Skype 2010) and (FlashMeeting 2010)) support informal communication in a synchronous fashion.

Collaboration refers to management of the manipulation of a shared work artefact by multiple people. This involves management of access includes locking and versioning of shared documents, based on check-in and check-out operations performed by users. Such systems include group (document) repositories and shared workspaces where artefacts are worked on asynchronously (Leuf and Cunningham 2001). Systems also allow the

synchronous editing of various kinds of documents. For examples of shared whiteboards and group editors see the BSCW system (Klöckner 2002) and VKB (Marshall and III 1997). More recently, Knowledge Construction for the purpose of stakeholder requirements analysis has been facilitated in real-time with flexible hypermedia environments (Shum, Selvin et al. 2006).

GoogleWave (Google 2010) is a shared space on the web which integrates chat, email and social networking to enable people to communicate and collaborate in real-time on rich format text (RFT) and other types of graphical content. The GoogleWave platform is extensible through 'gadgets'.

Collaboration platforms bring together basic communication ability, the joint manipulation of artefacts and some support for explicit coordination, the main example being Lotus Notes (Notes 2010).

## **Discussion**

Groupware technologies are leveraged most when they facilitate communication and collaboration for process participants who are separated in time and/or space. Being extremely flexible, user-driven systems, minimal process control is asserted for the work of individuals and groups. For this reason groupware technologies are more suited to the support of only the most fluid of processes. Their application to EKP is problematic. Lacking explicit process-orientation, they cannot maintain a consistent standard of procedure, nor lend themselves to process visualisation, analysis, optimisation or learning (Wang, Finch et al. 2009).

A truly flexible process support should support a wider range of process fluidity. This is the first occurrence of what is a common theme in the critique of existing work; the technologies often do not have an inherent weakness, rather the weakness emerges when applied to support work for which they were not intended.

A further problematic aspect of many groupware technologies is the lack of Web presence. This makes applications slower to deploy and more problematic for users to access and has contributed to a lack of take-up. The groupware technologies which have leveraged the Web tend to be communication-oriented (email and instant messaging) and lack rich user experience (Wang 2008). For example, the data modelling and computational support

provided by the GoogleWave gadget extensions are very limited as the change notification is very coarse (whenever a change to an attribute value happens, it can only notify the client that the data state changed, but cannot tell which attribute value has change). With such a limitation, more sophisticated groupware tools that need object-oriented modelling and object attribute level notification cannot be supported in the environment.

## 2.5.2 Explicit PAIS

The following section moves onto the description of tools and techniques which are explicitly process-oriented. To give context to the remainder of the discussion, the next section introduces some basic concepts and terminology.

### **Type and Instance Models**

Section 2.1 introduced the broad concept of meta-models and models. In BPM, it is useful to distinguish between type and instance models. A type model is a reusable definition of a process (Weber, Sadiq et al. 2009). The type model is subject to change (often large and infrequent improvements reflecting process optimisation, organisational engineering and compliance efforts.) This is shown in the middle layer of Figure 2-4.

The type model serves as a template from which numerous duplicates can be made (known as instantiation). Each type model can be instantiated an arbitrary number of times. Each instantiation creates an instance model which drives actual occurrences of the process. The scope for improvement in the instance model depends on the type model (domain requirements govern how much flexibility is built into the schema), but usually some mechanism facilitates ability respond to uncertainty/unknown conditions e.g. by navigating a particular path through model. The instance model therefore resolves any available flexibility opportunities. This is shown in the bottom layer of Figure 2-4.

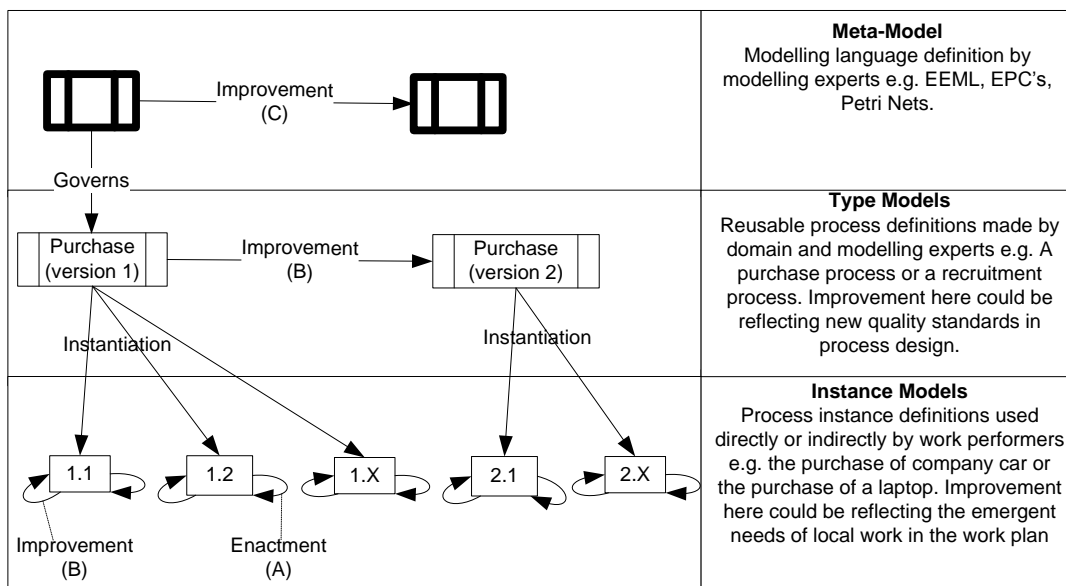


Figure 2-4 - The Interplay between Meta-models, Type Models and Instance Models

Figure 2-4 also draws upon Engelbarts 'ABC' work type framework (Engelbart 1992). 'A' type work (performance) occurs at the instance level only through enactment of instance models. 'B' type work (planning) occurs at both type and instance levels. 'C' type work (underlying way of working) occurs at the meta-modelling level. The following discussion will use the concepts presented in Figure 2-4.

### 2.5.3 Static Workflow

This original vision of workflow (also called production and enterprise workflow) addressed the automation of a business process, in whole or part, during which a workflow engine distributes documents, information or tasks from one participant to another for action, according to a set of procedural rules captured in a model (WfMC 2001).

Part of the workflow system is a process definition tool. Expert workers create the models workflow systems deploy as seen in Figure 2-5. From the work performer's perspective, coordination is achieved based on system based task 'pushes'. As the workflow engine steps through the model and a task is reached, work is distributed to a performer. This is often accomplished with some work allocation notification mechanism such as an inbox. Although model-driven, the complexity of the model is hidden from performers. For further information on work distribution mechanisms see (Muehlen 2004). The resource is expected to perform the allocated work and submit that work. The workflow engine is then



free to step through to the next task; linear runs are generated inductively during execution possible steps which preserve predecessor and successor relations are calculated (Pestic 2008).

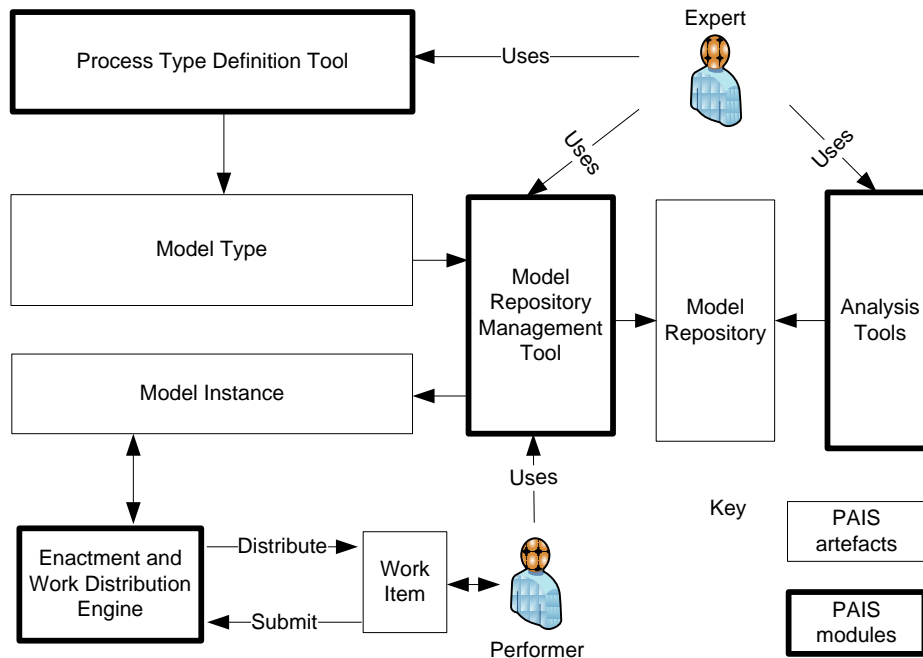


Figure 2-5 - Static Workflow Architecture

The setup costs of workflow systems are high. It is expected that these initial high costs can be recouped due to the heavy future usage of the system and the efficiency gains brought by its use (Reijers 2006).

Static workflow systems have enjoyed success because their scope and ambition are narrow. The approach relies on the execution of a strict script and their use is suited to highly standardised processes often found in industries such as banking and insurance (Jørgensen 2004).

## Discussion

Workflow systems automate highly predictable, routine processes. When their application is limited to such processes, their success has been reported (Georgakopoulos, Hornick et al. 1995). When applied to EKP, the highly prescriptive representations and deployment techniques employed generate high rigidity unsuitable to support the required individualism. It has been shown that when necessary process changes are not allowed by the system, it can result in action being performed outside of the system (Nutt 1996), (Cugola, G. 1998).

As a result two systems are not in alignment; the actual work is done in reality and later registered in the system in another way.

Again, as seen with groupware, this is only a valid argument when workflow technology is applied to process types it was not intended to support. Workflow technology is geared to cope with well-structured and pre-definable processes. As the original vision of workflow technology was not intended to support EKP a detailed investigation of it is of little relevance to this thesis.

#### 2.5.4 Flexible Process Support

By considering workflow and groupware, discussion till this point has described support located at the extremes of the specificity spectrum. It has been shown that the support provided does not meet the requirements of the modern enterprise and the EKP they employ. It is a synthesis of the two opposing schools of situated work and workflow work, which is regarded as the most promising approach to fill that void (Bernstein 2000), (Nutt 1996).

Flexibility requirements clearly dictate that formal organisational constructs such as process plans cannot be wholly rigid computational algorithms. The reconfigurable organisation (Levitt and March 1998), the organisational adaptation perspective (Galbraith 2001) and the real-time enterprise (Gartner 2003) all acknowledge the importance of the ability to quickly adapt. Therefore, recent focus has been on supporting flexible work activities with flexible artefacts that are nonetheless driven by an underlying notion of a work plan (Klein, Dellarocas et al. 2000), (Daoudi and Nurcan 2007). The assumption is that many processes can be driven with a script, if that script has the ability to adapt to the actual work practice and to changing conditions.

#### **From Performers to Knowledge Workers**

PAIS have addressed the need for flexibility with the move towards user-centric systems where the strict roles (expert for modelling and performer for enactment) and model phases (modelling followed by enactment) established in static workflow systems are broken (Lillehagen 2004), (Faustmann 2000):

- Performers are empowered not just to perform work, but to think about how that work should be performed. In short, the knowledge worker is not only a work performer but a work designer.
- The traditional modelling phases established by workflow become intertwined, with rapid shifting back and forth between modelling and enactment.

In Section 2.2, a traditional approach to BPM was described in terms of the ‘ABC’ work type framework (Engelbart 1992). A modern approach to BPM is delivered by flexible process support efforts which changes this traditional way of working. The same work is performed but the difference lies with who performs it. ‘A’ level workers (performers) are empowered to move up the hierarchy and perform ‘B’ level work in the form of process articulation (planning) and interpretation (analysis) in an ongoing fashion. Throughout the work endeavour, users will continually switch between ‘A’ and ‘B’ level work as required. As this is still delivered in a PAIS, C level work is not affected.

This approach relies on contribution from the knowledge worker in the form of a solution to a design problem. The design problem in this case is the formulation of an instance model; configuring available means to reach desired ends whilst obeying laws. The artefact generated must satisfy particular domain requirements.

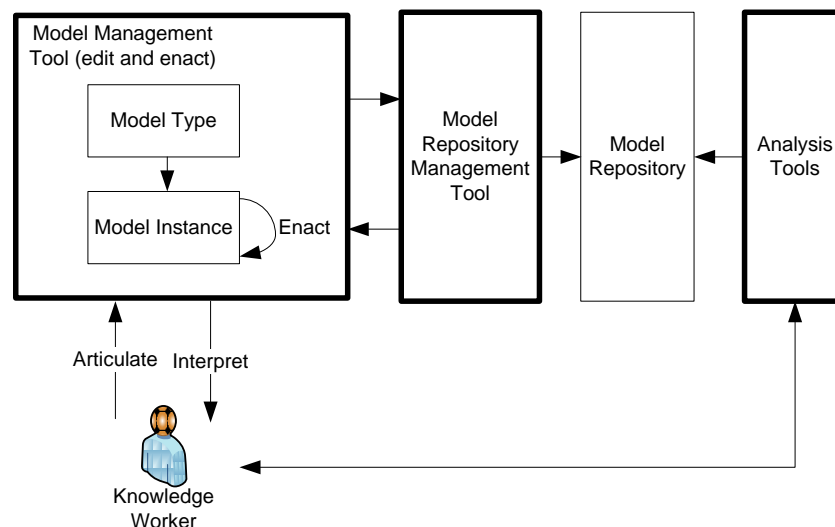


Figure 2-6 - Model-driven Approach to Process Support

A key question that the empowerment puts to BPM research community is how users can be successfully leveraged to create, analyse and adapt process structures. The model in

Figure 2-6 has been achieved in various ways, some maintaining certain properties of workflow, others embracing more open system design.

### **Burden of Empowerment**

The charge of process planning and composition is by no means trivial. It has been suggested that although ad-hoc design and changes of work plans are required, ordinary end users cannot be trusted to carry out this work (at least not in a controlled way). This claim is based on the view that process modelling is difficult, time-consuming and requires expert knowledge (Jablonski 1994). Process composition is often:

- *A Complex Task* - Models can be intricate and interconnected. Holistic models place extra burden on the maintaining a coherent whole, alignment between the various aspects.
- *An Unstructured Task* - Interconnected factors constrain and inhibit how a process should be designed; change can be described as a multi-criteria decision problem where the evaluation criteria of the change may be multiple and conflicting. Compounding the situation, participants make design choices with ambiguity, uncertainty and unavailability present in the available information (Oquendo, Papamichail et al. 2000).

Furthermore, time constraints and cognitive limitations in the face of large and complex processes (Courtney 2001) will impact on the quality of process composition. For these reasons, some researchers challenge the feasibility of articulation by end users, arguing that they cannot be expected to create and adapt models correctly (Jablonski 1994).

However, there are many successful systems which rely on user articulation. For examples of user articulation in open systems we need look no further than flow of information on the Internet. The distinction between providers and consumers is becoming problematic. A central theme of the modern Web is that users have no longer a passive role, but they are generating content in many different ways e.g. blogs, comments, photos and video.

Can the approach of user participation be successful for more formal process-oriented business applications? Bansler and Bødker (Bansler and Bødker 1993) reported on the use of DFD's in the performance of a domain analysis task by novices. They found that the techniques were applied pragmatically and subjects reported that the diagramming technique was useful. More recently, the Rapid Knowledge Construction approach of

(Selvin and Shu 2002) demonstrates how stakeholders can capture and analyse formal and informal information collaboratively. For a more process-oriented example, the empirical results of the EXTERNAL project shown that semi-skilled user were able to build, analyse, adapt and enact process and enterprise models (Lillehagen, Krogstie et al. 2002) and (Wang, Finch et al. 2009).

### **Key Challenge**

It is essential for the success of the approach that high quality process models (type and instance) are generated. The primary goal of the model is to help fulfil the goals of the user and group in their situated action (Krogstie and Jørgensen 2004). The possibility of modifying process definitions is fundamental in order to comply with the progressive elicitation of process structure that EKP demand.

This is obviously a balance since one should not be able to change parts of the model that it is defined as very important for the organisation to keep stable (Krogstie, Sindre et al. 2006). Process innovation can produce negative impacts on businesses if they are not managed properly. Organizations are more than ever are striving for maturity in the processes they implement. This is often due to both the abundance of legislation and the potential of mature processes to gain and maintain competitive advantage. Compliance is defined as ensuring that business processes, operations, and practice are in accordance with a prescribed and/or agreed set of norms. This issue considers the semantic quality of models; the degree of congruence between the model and the domain its represents (Krogstie, Sindre et al. 2006).

A balance which must be mastered emerges; process flexibility and process control (Sadiq, Orłowska et al. 2005). Providing a workable balance between flexibility and control is an enduring organisational challenge (Sadiq 2005). This balance can be described as having the ability to bend, but not break process compositions. Organisations that understand this balance can prepare so control can be maintained over the changing situations. How organisations can go beyond grand statements of intent with regards to mature processes and realise that intent in day-day operations is still an open question and a main theme of this research.

The following section looks at flexible systems with a particular regard for how they approach this balance necessary for EKP. In addition, other critical areas are identified.

## 2.5.5 Conceptual Approaches for Achieving Process Flexibility

Knowledge workers consume and resolve available flexibility opportunities in instance models. Competing paradigms have emerged which provide flexibility opportunities. Each paradigm allows the deferment of decisions (which can be based on predefined rules or user decisions) regarding the exact control-flow or instance model composition when more information is available. Thus, planning and execution is closely interwoven to allow for a more agile way of planning. The following discussion is based on Schonebergs classification (Schonenberg, Mans et al. 2006).

### **Flexibility by Enumeration**

This approach attempts to include all possible execution scenarios within the type model to cater for all possible contingencies. This flexibility is also known as flexibility by design and requires a deep insight into the application context. This is most often achieved through conditional branches. A particular path can then be tread in each instance model. For an overview of the various patterns of splits and joins see in workflow articulation see (Aalst, Hofstede et al. 2003).

A particular style of this flexibility is achieved not through paths, but through exceptions. Exceptions in PAIS are any temporary deviation from the normal prescribed flow of the work. This requires exceptions to be anticipated and predefined during the design of the type model; the designer analyses the process model in order to identify its potential exceptions. Once the potential exceptions are identified, the proper handling routines can be included in the type model. For a full description of exception in workflow see (Adams 2007).

There are two drawbacks to this approach. The first is that the model is greatly complicated by the presence of paths and exception handling data, increasing the expertise required to use it. Secondly, anticipating all cases and planning for all possibilities in many domains is difficult. This reduces applicability to a narrow process base which excludes EKP.

### **Flexibility by Under Specification**

Structure emerges as the work progresses, not by navigating a path through an existing model, but by building all or part of the model. This approach removes the great complexity of having to deal with the definition of all possible instances in one type model. The more structured realisations of this approach recognise that many processes are routinely configured, that is the same re-usable process space is used but different compositions within that process space. Four main techniques for under-specification are described and have been adapted from (Weber, Sadiq et al. 2009) and (Aalst, Adams et al. 2009):

- *Multi-Instance Process* - Multiple instances of a task can be executed during runtime; the number of task instances can be determined at runtime. This pattern offers the least amount of freedom during run-time. It allows users to defer the decision on how often a specific task should be executed to runtime, while the task itself needs to be predefined.
- *Late Selection of Process Fragments/Late Binding* - This pattern offers slightly more flexibility by deferring the selection of the implementation of a particular process task to run-time. Prior to execution, at design time, only a placeholder activity has to be provided. The concrete implementation is selected at runtime from a set of predefined fragments, replacing the placeholder. Thus selection is made through predefined rules or user decisions
- *Late Composition of Process Fragments* - At design time a set of process fragments is defined out of which an instance model can be composed at run time on-the-fly. This can be achieved by dynamically selecting fragments and adding control dependencies on-the-fly whilst adhering to constraints. The modelling of the placeholder activity must be completed before the process can be executed.
- *Late Modelling of Process Fragments* - Parts of a specification are left undefined at design-time using placeholders and are modelled during runtime for each instance model. This can be achieved by dynamically selecting fragments and adding control dependencies on the fly whilst adhering to constraints. The modelling of the placeholder activity must be completed before the process can be executed. The difference between this and Late Composition of Process Fragments is that the user need not be limited to the pre-defined set of fragments in model design.

The last two techniques can vary with the amount of structure imposed on the user (Weber, Sadiq et al. 2009):

- *Whole process, unconstrained* - The whole process schema may be defined during runtime without any constraints on composition.
- *Part process, unconstrained* - Parts of the process are predefined, while parts can be flexibly defined during run-time. Parts which can be defined at runtime do not have constraints on composition.
- *Part process, constrained* - Parts of the process may be well defined, whereas other parts of it can have a prohibitive number of execution options that can only be determined at run-time based on tacit know-ledge (e.g., a particular combination of case features or the expert opinion of a knowledge worker).
- *Whole process, constrained* - Any part of the process can be modified. The modification is under the influence of constraints.

### **Flexibility by Change**

Flexibility by under specification essentially provides some initial structure with open ‘hot spots’ which can be resolved by end users. Flexibility by change provides initial structure but that structure is adaptable. So that structure can be considered a tentative starting point which can be changed to suit the needs of the instance. The focus of this study is how to cope with emerging business needs at the instance level. As the type model can change, a research issue has been in how to deal with active instances which are adhering to an outdated type model. This is called the dynamic change problem. This is not the focus of this research, interested readers are directed to (Ellis, Keddara et al. 1995) for further discussion.

Different approaches have been used to realise the different approaches to flexibility described above in concrete systems. The three key approaches examined below are flexible workflow, interactive models and integrated systems.

### **2.5.6 Flexible Workflow**

One approach to flexible process support is to make workflow based technology more flexible. Flexible workflow solutions address the need for individualism through instance model design and execution. A huge research effort has addressed this goal over the past fifteen years (Cugola 1998; Nutt 1996; Gonzalez 1999; Kammer, Bolcher et al. 2000; Kwan and Balasubramanian 2003).



Technologies have been classified according to their primary research interest (of flexibility by enumeration, under specification and change).

### **Systems for Flexibility by Enumeration**

Examined below are key systems related to the handling of exceptions. This provides an important perspective on user involvement and balancing flexibility and process control.

OPERA is an academic prototype (Hagen and Alonso 2000). The workflow engine will step through a standard workflow definition. In the event of an exception (when domain requirements fall outside the standard model), enactment is stopped and control is passed to the predefined handler for that exception. Exception handlers are organised hierarchically. If an exception arises and no handler is found during navigation of the hierarchy, no further support can be provided by the system. OPERA therefore takes a minimal user intervention approach.

ADOME (Chiu, Li et al. 2000) adopts a slightly more empowered approach to user intervention in the event of an exception than OPERA. Workflow ‘templates’ can be used as a basis for a workflow at design time. Declarative (Event Condition Action rules) and procedural (meta-process) techniques are used to define exception handlers. This database is added to as useful exception handlers are identified. The exception handling service is automated. ADOME differs from OPERA in that if no exception handler is suitable full or partial control is passed to human actors. Some support for insertion of process fragments for placeholders is available at execution time.

MILANO (Agostini and Michelis. 2000) adds more user control still by allowing the performance of ‘jumps’ such as redo, loops, skipping an activity, parallelisation and sequentialisation. Control over which resources are allowed to perform which jumps is achieved through organisational policies. Furthermore, MILANO supports user negotiations in exception management process with an integrated conversational tool and the availability of contextual (historical) data. Once the strategy is formulated it can be verified for correctness (for the current and future usages of the strategy) and then is applied manually to the process.

Contemporary work on exception handling is provided in (Adams 2007). Adams conceptualises and implements the Worklet Service which leans towards a late binding

approach. Instance models are dynamically built to cater for individual instance needs. Placeholders may occupy various points of procedural workflow definition. Placeholders represent unspecified parts of the model, which are to be determined by the Worklet Service at run-time. Such placeholders are substituted with a 'Worklet' from the available set. Worklet choice is resolved by using context data along with selection rules. For comprehensive coverage of commercial and academic exception-oriented workflow systems see (Adams 2007).

In (Gottschalk, Aalst et al. 2008), a configurable process model makes use of variation points to capture the differences among the process variants. Business analysis can then configure this model to their specific needs by choosing the proper process variant for each variation point. This approach provides an alternative to designing process models from scratch. Meanwhile, it can foster the systematic reuse of proven or common practices in a given industry domain. A core feature of configurable process models is the explicit representation of variation points and their variants. A variation point can be indicated in different ways, e.g. it could also be a special activity.

## **Discussion**

This way of working relies on the predefinition of all possible process scenarios and is geared for when flexibility is required in only the minority of work. Flexibility is inhibited as user involvement is not ingrained into everyday work but only on those occasions where a standard way of working is unsuitable. Even then, the user has limited number of predefined options in handling this situation; initially underspecified models cannot be incrementally developed. The reactive involvement of users is suitable only for when attention is required in a small proportion of cases.

These properties make the approaches application to EKP problematic as a fundamental property of EKP is an inability to predefine all possible execution scenarios. Furthermore, user collaboration opportunities are rare, such as in the joint creation and inspection of plans between process participants. As EKP are frequently performed by distributed teams with multiple stakeholders, team working is becoming increasingly important.

## **Systems for Under-specification and Change**

It is useful to categorise these systems by their operational abstraction.

### **Procedural Workflow Systems**

Mobile (Jablonski 1994) provides services for both flexibility by enumeration and flexibility by change (through modification of the workflow class definition). However, the authors claim that although ad-hoc changes will occur, ordinary end users cannot be trusted to update models in an uncontrolled way. This claim is based on the view that workflow modelling is difficult, time-consuming and requires expert knowledge. This way of working thus maintains traditional roles (professional user for work management and ordinary user for work performance) of workflow.

Inconcert (Sarin 1995) is a commercial workflow system which strongly leverages human agents to achieve ad-hoc work processes. Templates are provided which serve as a starting point for instance models. Templates can assert a varying degree of structure; four strategies for template deployment are used. First, a new instance can be created by replicating an existing process template. Second, it is possible to create a new instance based on a previously changed process model. Third, an ad-hoc instance can be initiated by specifying a sequence of activities. Fourth, a new instance can be initiated as a 'free routing process', where the created instance is initially empty, and its actual routing is created on-the-fly. Initially the instance is equivalent to the template used. Process participants are allowed to modify instance models on-the-fly through local articulation at runtime. Instance-orientation also simplifies the modelling language and management. This way of working has strong scope for human judgment and collaborative working through a shared workspace.

Endeavours (Kammer, Bolcher et al. 2000) is a comprehensive workflow solution which combines a number of techniques for exception detection, avoidance and handling of exceptions. In doing so the approach moves into instance and type evolution. Emphasis is on support for the user in this process once process execution has begun. Hierarchical models with varying levels of process guidance (loose to strict models) can be asserted. On the fly work composition is supported by libraries of re-usable fragments and components, integrated support for participant communication and collaborative working. The collaborative working approached is 'divide and conquer' where individuals take responsibility of their own parts of the process.

### Declarative Workflow Systems

It is widely recognised that the ideal (prescribed process) and the actual work which takes place are often different. This does not necessarily indicate that the work occurred outside of system control but rather that, the original prescribed plan was modified in some way. This brings into question the value of the fully prescribed plan and if they try to anticipate too much. If the prescribed model is often changed, does this not indicate wasted effort in its formulation?

At the other end of the operational abstraction spectrum we have systems which employ declarative or descriptive languages. It has recently been recognised that declarative approaches have a good potential for achieving process flexibility as they do not require a full specification of the process to be executed to be made. The artefact which drives work is in the form of constraints which capture the boundaries of possibility of work design and process behaviour, rather than a fully formed plan prescribing precisely how participants are going to perform their work (Pesic and van der Aalst 2006). This is a flexible approach as the possible execution alternatives are only implicitly declared which can be any behaviour which does not violate constraints is allowed (Pesic 2008).

Every notation highlights some kinds of information at the expense of obscuring other. As the language is restricted, it is possible that focus is on those aspects which can be satisfactorily represented in the model, whilst other equally important aspects are ignored. It is well recognised that a lot of relevant business knowledge is not captured in traditional procedural process model. Evaluation of various modelling grammar consistently identify certain deficiency in declarative information (state law and state space) (Rosemann, Recker et al. 2006).

This representational weakness results in incomplete or informal models with regard to organisational compliance requirements. It has been recognised that plugging this deficiency can help process support systems move towards a balance between control and flexibility (Sadiq, Orłowska et al. 2005).

Research in this area is currently a 'hot' subject in BPM. There is a vivid interest among practitioners and commercial software vendors in the confluence of business rules and business process modelling (Debevoise 2005). Addressed here are systems that use

declarative information (business rules) to constrain the final form of the plan (work which will occur).

Wholly declarative approaches to flexible process support have emerged. The key work in this area is the DECLARE system (Pesic 2008). Processes are defined in a declarative way using a predefined (but extensible) graphical language. DecSerFlow (Aalst and Pesic 2006) is predefined set of constraints focusing on the task aspect to establish constraints such as 'if X is executed, at some point in the future Y should be executed'.

The resultant models are deployed and executed directly (rather than used to drive the design of a procedural artefact.) This approach is less demanding for the performer. With a given model in a given state the user is presented tasks which can be legally executed. As work is performed, gradually constraints will be satisfied. The set of possible execution alternatives are implicitly derived from the constraints. The process can finish in a legal state with any execution alternative that satisfies all constraints. Models can be modified during execution, although this would require some skill, which usually means changing the set of execution alternatives.

The major weakness of this novel way of working is a sense of forward planning or retrospection (what work has happened before and what work will occur in the future) cannot be established by the user. The user cannot easily visualise each task instance (and the resources and data associated with it).

### Hybrid Workflow Systems

These are techniques and tools which combine both procedural and declarative information. Mangan and Sadiq develop key work in this area which is realised in the Chameleon prototype (Sadiq, Orłowska et al. 2005). The way of working is centred on the idea of 'pockets' of flexibility. A traditional procedural model is modified to incorporate 'pockets' of space where process compositions can be made 'on the fly'. Initially 'pockets' can be empty or partially defined. Full specification of a 'pocket' is made at run-time before their enactment and may be unique for each instance model.

Each 'pocket' can contain traditional procedural information as well as declarative information. The procedural content drives actual work. The declarative content restricts the form of the procedural content by articulating qualities to which it must adhere i.e.

initially the sub-workflow definition of a task may have declarative content only which is used to check the quality of eventual procedural content. A set of predefined constraint types are defined (serial, fork, order, include and exclude) to articulate qualities which the procedural model must adhere to. The predefined constraints focus on the task/control flow aspect. The approach is workflow-oriented as the configuration opportunity above is performed by a skilled user/system administrator. The work is important as it demonstrates an approach for the validation of a set of specified constraints.

WAM (Faustmann 2000) is a flexible workflow system which uses detailed hierarchical process description to implement an approach where workers can perform allocated work, delegate work or create a sub-process to plan how work should be performed. Its successor, LAW (Faustmann 2000), can restrict the form of the work which is ultimately performed by adding a controlling element to this way of working. Associated with each complex task is the participants' authority to change the model. Thus the approach combines enforced scheduling of tasks with freedom for the end users where allowable. The permission is based on:

- *Can* - The process is a recommendation, so the actors may use another method.
- *Strict* - The specification is mandatory, and no changes are allowed.
- *Must* - Specified steps must be performed, but new subtasks may be added.

The KISS (Feldkamp, Hinkelmann et al. 2007) approach is described as 'agile business process management'. To cope with the requirements for flexibility, consistency and compliance business rules are used alongside processes. Traditional procedural models are given in visualisable and executable form. Business rules are used in process execution for resource allocation, constraint checking and decision making (branching decisions).

The FREEFLOW (Dourish, Holmes et al. 1996) system implements a rich constraint based process modelling formalism. By leveraging task dependency relationships with an extended task state model (inactive, active, disabled, enabled, ready and pending) it can be asserted for example that an activity can enter a specific state only if another activity is in a certain state. Components subscribe to notifications about events that occur in other components. The FREEFLOW system allows users are able to override constraints.

Tucupi (Wainer, Bezerra et al. 2004) uses a constraint based workflow definition with rules involving (1) pre-conditions that must hold before an activity can be executed, (2) post-

conditions that must hold after an activity is executed and (3) par-conditions that must hold in general before or after an activity is executed. Similarly to FREFLOW, the conditions are defined based on the states of other activities. Workflow is defined as a set of pre and post conditions of activities. Activities that have all pre conditions met are dispatched by the workflow engine. Again constraints can be overridden by authorised users. Based on the given constraints the system can help the user plan a workflow.

GPSG (Generalized Process Structure Grammar) (Glance, Pagani et al. 1996) use constraints to describe the set of process possibilities allowed, which collapses onto one choice only when the time for action arrives. Constraints establish dependences among tasks and documents. This interweaving of task and document structures enhances the expressiveness of the language and the richness of the coordination functionality. The task centred constraints focus on what activities are required to achieve a goal and any temporal dependencies between those tasks, for example “task A should finish before Task B”. New constraint rules can be added dynamically during the execution of the process.

AdaptFlow is (Greiner, Ramsch et al. 2004) driven by a traditional procedural model. The models can be modified at runtime (i.e. add, delete and change the sequence of tasks) both at the type (dynamic evolution) and instance levels (ad-hoc changes). These changes are made through manual intervention. Each adaptation must be confirmed manually by an authorised user before it is applied. The great importance of this work is the attention paid to maintaining semantic correctness and preventing semantic conflict using rule based process adaptations. The instance is modified based on the rule which has been triggered.

In AgentWork (Muller, Greiner et al. 2004), the procedural workflow instance can be modified by the manual interaction (add task, remove task) based on events and ECA rules. A weakness is the scope in this approach for rules to generate incompatible actions. When conflicts emerge manual intervention is required for its resolution.

(Ly, Rinderle et al. 2006) present a technique for manual process composition at runtime. A pool of activities is available. From this pool, users select activities according to the needs of the particular case. Decisions are made by each user for each case. Constraint rules can be specified to control the selection of activities. The constraint specification approach is limited to preventing certain combinations of activities being selected.

## Discussion

Various procedural workflow systems were presented which allow the modification of instance models on-the-fly. These systems encourage a more proactive user with often manual interventions. A problem stemming from these added powers of flexibility is the need to support end-users in the selection and modification of the process. This has not been well addressed by the procedural workflow systems described above.

Maintaining the semantic correctness of models is a key goal in flexible process support (Sadiq, Orłowska et al. 2005). Practical guidance for the modification of process models is not forthcoming (no methods or tools are used to indicate if the modification maintains the semantic correctness/compliance of the model). The resultant flexibility can be described as creating brittle processes, that is to say, easily broke as change is enabled but no control over that change is provided.

Declarative and hybrid workflow systems begin to explicitly address a critical requirement for EKP of balancing process flexibility and process control. Opportunities for change or configuration are any that fall within a valid space. These approaches are often aimed at expert users and are difficult to activate. Furthermore, these systems are characterised by a lack of evaluative studies, it remains uncertain if end users consume declarative and hybrid model-driven approaches.

Various user support strategies can help users interact with graphical models and any flexibility mechanisms they possess. As flexibility mechanisms become more complex and the burden on user greater, the issue of user support has become critical. User support for model interaction has come in different forms:

- *Change Patterns* - Change patterns are high level process adaptations. Common changes to the model (e.g. insertions and reordering of tasks which are often made up of multiple low grained primitives) are available to the user as a single operation. Such change patterns hide the detail of the operation they implement reducing the complexity of model interaction for the user. See (Weber, Reichert et al. 2008) for the definition of seventeen change patterns.
- *Wizards* - Wizards essentially guide the user through the process of making a choice regarding model composition where that choice is governed by rules or constraints. Wizards hide the detail of those rules from the user and establish the choice by asking the user a series of questions (knowledge workers can configure



process models by answering interactive questionnaires). Questionnaires are captured in questionnaire models. Answers are used to automatically generate an individualised version of the configurable process model.

What is common amongst each of the techniques described above is that they are consumed by the individual. To survive and thrive, enterprises will have to work toward adapting their processes not only in real-time (to minimise downtime and delays in work progression) but also with group consensus (as multiple stakeholders in process design is common). Removing the delays that slow down the work progression can be achieved through real-time response and group consensus in the identification of problems (inadequacies in process design) and response to those problems (agreement in process adaptation) (Wang, Finch et al. 2009). This instant process-oriented collaboration facility is missing from the above systems. Some systems do include some groupware features, like shared workspaces for documents, but not extensive support for e.g. negotiations and synchronous direct manipulation of process models (Wang, Finch et al. 2009).

### 2.5.7 The Interactive Modelling Approach

The discussion so far has focused on workflow-oriented systems. The interactive modelling paradigm addresses similar goals as workflow technology. The interactive modelling approach seeks to coordinate the activities, data and resources which comprise an organisation and its processes (Jørgensen 2004). The approach is centred on the Knowledge Model which is a visual representation of enterprise aspects that can be activated (created, viewed, traversed, analysed, simulated, adapted and executed) by industrial users (Lillehagen, Krogstie et al. 2002). Knowledge Models form a central part of the user's day to day working environment; there is a co-evolution of the Knowledge Model and the domain.

The promise of workflow-oriented solutions for coordinating organisational processes drew strong criticism of their rigidity of their work representations; they often rely on processes which can be both predefined and articulated in detail. The interactive modelling approach shifts away from a closed and automated way of working to systems which are more manual and open that aim to help people carry out work in the way they want to carry it out (Jørgensen 2004). Plans are articulated to a formality and depth which is useful for the

work at hand. The ready tangibility of Knowledge Models supports the capture unfolding and dynamic business forms and processes (Lillehagen, Krogstie et al. 2002).

A key particular goal of the interactive modelling approach is the coordination of teams which may be distributed in time and space (Wang, Finch et al. 2009). The described 'push' of workflow-oriented systems is replaced by the 'pull' of interactive models in addressing this concern. The Knowledge Models form a central artefact for coordination and collaboration amongst team members. Instead of a hidden model used for automated work distribution, actors are drawn into the model to view, learn and develop the process as a whole.

Workware, (Jørgensen 2004) is a prototype system which realises the principle concepts of interactive models. Graphical modelling at the instance level is heavily promoted to capture the unique conditions of work. Instance modelling removes the great complexity of having to deal with all possible variations in one model. The approach is heavily reliant on user modelling skill as all interactions with the model are manual. Workware lacks group working capability and fails in the need for fast roll out of the application to support temporary cooperation between networked organisations.

XCHIPS (Wang, Finch et al. 2009) is a cooperative hypermedia based modelling prototype which implements an interactive modelling approach to address flexible process support. XCHIPS strongly promotes and addresses the need for real-time collaboration in the creation, analysis, inspection and modification of work plans.

XCHIPS forms a multi-user distributed modelling system based around visual hypermedia structures which can be more or less structured. The meta-model and computational supports are readily extensible and so can be tailored for specific domains. The system has been applied successfully to process-centric domains such as meeting support (Wang, Haake et al. 2004) and project management (Wang, Finch et al. 2009).

Goal-driven process articulation is achieved through hierarchical models. If sub-content is defined for a task then that task becomes a goal and its sub content is charged with achieving that goal. This way of working does not readily facilitate the articulation of goals which can be checked objectively. Adaptivity is achieved (as models can be modified at any time), but in crude fashion as it is devoid of any constraints. The result is a difficulty in

ensuring the semantic quality of the models at a fine grain. This way of working is unsuitable for many of the compliance aware organisations which comprise the modern enterprise landscape.

Both of the described systems facilitate process articulation with varying degrees of structure; from traditional formal process models to informal artefacts such as collections of tasks, annotations and drawings. Informal articulation of plans allows a more direct representation of the user's mental model. In this case, the system serves falls out to a whiteboard.

### **A Comparison of Interactive Models and Workflow Approaches**

To gain a thorough understanding of the interactive and workflow process support paradigms, it is useful to compare them from the knowledge worker perspective. This is summarised in Table 2-1.

Table 2-1 – A Comparison of Interactive Models and Workflow Approaches

| <b>Function</b>  | <b>Static Workflow</b>  | <b>Interactive Models</b>  |
|--|---|--|
| <i>Scope of role</i>   | Work performer  | Work designer/performer/analyst  |
| <i>Work primed for</i>   | Predictable   | Unpredictable  |
| <i>Work plan</i>   | Specified in advance by expert  | Ongoing specification by knowledge worker                                  |
| <i>Plan enactment</i>  | Automated   | Manual and automated   |
| <i>Plan visibility</i>   | Limited – no means of viewing work items that are not intended to be handled by this resource | Full – tangible processes through up and down stream process visualisation |
| <i>Process interactivity (e.g. planning of new work items)</i> | Limited   | Full   |
| <i>Method of task allocation</i>                               | Personal mailboxes, work lists  | Graphical model, derived work lists  |
| <i>Activity assignment method</i>                              | Push, negotiate   | Pull, negotiate  |

### **Discussion**

The support provided by the interactive modelling approach is in close alignment to those required by EKP. This approach forms key ideas for the remainder of this thesis. However, there is still a lack of engineering research which develops and validates the interactive models as a design approach (Jørgensen 2002). This is highlighted particularly with regards to the integration of modelling techniques which can address compliance requirements and how the approach can better leverage the rapid deployment and ease of access offered by the Web infrastructure.

### 2.5.8 Integrated Systems

Until this point, the majority of the systems described can be characterised as occupying a single point in or at best limited zone of the specificity spectrum (Bernstein 2000). Such specialisation (focusing on very specific needs of very specific types of processes) strongly contributes to the successes they have achieved. However, many processes cannot be readily characterised as they have shifting and diverse requirements in terms of the support required. This is the case with EKP. For example, process design might evolve, e.g. moving right on the spectrum to accommodate unforeseen exceptions, or moving to the left as practices become more structured as learning and experience grows. Furthermore, an organisation may have multiple processes with varying support requirements which are supported with a fragmented toolset.

Integrated coordination support across varying process types, covering a wider area of the spectrum is an important challenge. The process flexibility so far discussed is within a single approach. A second type of flexibility is then to apply multiple approaches, to offer the user a choice of support techniques where the most relevant for current needs can be selected (Aalst, Adams et al. 2009). Integrated systems have not yet materialised in working software systems to a great extent but are beginning to emerge. The following discussion targets systems which can provide a wider degree of specificity.

EXTERNAL (2003) is an EU fifth framework program on information technology support for new ways of working. The aim of the project was the development of information technology to support temporary cooperation between networked organisations in knowledge intensive industries. The need for temporary integration across organisational boundaries remains a current focus (Collins, Ketter et al. 2010). The project focused on the development of an approach to dynamic planning and work management for distributed teams. Team members should be able to plan, simulate and perform tasks with changes in plans and resource allocations. More concrete requirements which the project identified included the need for joint planning and analysis of business processes, joint execution of the processes, coordinated access to shared documents and joint support for the detection of problems and adaptation of work processes (Wang, Finch et al. 2009). These requirements were addressed with an integration of project management, workflow management, information management and cooperation. The EXTERNAL infrastructure integrated several of tools to realise a model-driven way of working including METIS (Lillehagen 1999), a generic tool enterprise modelling tool with strong visualisation capability,

XCHIPS (Wang, Finch et al. 2009), described above, SIMVISION (Kuntz, Christiansen et al. 1998), a simulation tool addressing task and resource aspects, WORKWARE (Jørgensen 2004), described in Section 2.5.7.

YAWL (Aalst, Aldred et al. 2004) is driven by the ambition to be able to provide a comprehensive support for most workflow patterns while maintaining a relatively simple language. YAWL is built as a traditional workflow management system, i.e., ordering of activities is defined in the traditional 'control-flow' manner. YAWL is an open system with a Services-Oriented Architecture (SOA) for ready incorporation of different process-oriented services and has been used as a central hub to implement the Flexibility as a Service (FAAS) concept (Aalst, Adams et al. 2009). For example the previously described Worklet and DECLARE prototypes have been integrated into the YAWL system to provide choice of tool for work situation.

The ActivityFlow specification language described in (Liu and Pu 1997) divides workflows into different types at design time (including ad-hoc, administrative or production), and provides an open architecture that supports interaction and collaboration of different workflow systems.

The tool proposed in (Bernstein 2000) seeks to support the entire range of the specificity spectrum by providing support for well-specified and routine to highly unspecified and situated processes. Four zones of support are identified. The first zone does not specify the process, rather a shared space is provided for informal shared work lists, document management and communications. The second zone increases specificity as here users can add constraints such as deadlines to process specification. The third zone allows users to add end states of work. With an incorporated AI planner, the system can now suggest ways of completing work which keeps with specified constraints. The user can follow the plan or create one of their own. The fourth zone acts more like traditional workflow and provides imperative scripts.

## **Discussion**

Integrated systems begin to meet the challenge of process diversity essential for the flexible support of EKP. This is accomplished not with the creation of a 'one size fits all' way of working, but by considering how multiple ways of working can be effectively be integrated under the banner of a single coherent system. The user then selects the most relevant

support for the work at hand. This approach is very promising for the requirements of EKP whose support needs can frequently drift up and down the specificity spectrum. Each described system currently fails to take advantage of the Web technologies for rapid deployment and ease of consumption. Dominant desktop applications can be cumbersome to deploy (for administrators) and consume (for users) which discourages take up and slows down inter-organisational connections.

## 2.6 Web Service Composition

The research being performed in the field of Web Services is highly relevant for this thesis. Automated applications from service providers are offered as services that can be composed and consumed by other parties. Individual web services are capable of providing some functionality on their own but the greater value is derived by combining several web services to establish more powerful and useful applications.

A key work in this area is the large scale Web Service-Oriented Architectures for All (SOA4ALL) research project (soa4all 2010). Forming part of the European Seventh Framework Programme, its main goal is the creation of a framework and infrastructure to realise a comprehensive service delivery platform where Web-enabled organisations can offer and consume services (Rey, Cekov et al. 2008). The delivery platform provides a central place for users to search and compose Web Services to create, modify, annotate, share, execute and analyse processes. The project extends the scope of processes to include both human tasks and Web-based software services.

The project demonstrates how IT solutions can be made readily accessible to non-technical users, which is a key requirement of EKP. This is achieved several ways. Firstly, an easy to manage, lightweight process modelling seeks to lower the barrier to entry for process modelling. Secondly, the project makes access to the service simple with a Web-based front end. The delivery platform is a Rich Internet Application (RIA), capable of providing a rich user experience more traditionally associated with desktop applications.

Process control is only partially achieved with limited scope to articulate low level semantic requirements on process composition. Synchronous collaboration is also missing from the framework.

## 2.7 Passive Process Capture

The discussion so far has centred on the active use of models; model-driven techniques and technologies which directly deploy and activate models in a PAIS. Models are also used passively for purposes such as IS development, human-sense making and communication (Krogstie 2008). These research efforts have strong relevance to this thesis.

It has been established that the mix of procedural and declarative information in process articulation can contribute to achieving a balance between process flexibility and process control. In process support, use of declarative modelling techniques is in its infancy. We can look towards the more established handling of constraints in IS development and process modelling for learning opportunities.

### 2.7.1 System-Oriented Models for IS Development

Information system development can be viewed as a transformation from perceptions of a real-world system to a working information system which is a representation of that real system. The transformation is model-driven. The models progressively become more and more detailed, as implementation related factors are taken into account:

- Early stage models are high-level, human-oriented analytical conceptual models. During the initial phases, the models developed are abstract, focusing on the external qualities of the system.
- Mid stage models are system-oriented, detailed design models.
- Final stage model is machine-oriented, executable software.

The models the IS is based on will change infrequently and usually for major modifications, rather than the rapid and small day to day changes described in this work. It is recognised that compliance considerations are vital. Compliance has impacted on this way of working as constraints or rules are represented in conceptual models, realised in design models and implemented in software. Constraint capture and representation is common in object-oriented analysis and database design. Rule statements restrict valid states of an IS by referring to entities in model e.g. an E-R diagram often takes role of the model with a business rule representation constraining valid instances (states). Two key approaches to IS development are now described.

## UML

UML (UML 2010) is the industry standard software modelling notation. UML is an object-oriented approach using a collection of complimentary, graphical notations (e.g. class diagrams, activity diagrams) to create a rich picture of subject domain phenomena.

Constraint handling in all stages of development can prevent the system entering an illegal state. Similarly to procedural models in BPM, the reality is that a graphical model using predefined constructs, such as a UML class diagram, isn't sufficient to make a precise and unambiguous constraint specification. To address this, UML models have been supplemented to articulate constraints with several strategies:

- *Annotation* - Supplement those diagrams which don't otherwise have the ability to represent the information:
  - *Unstructured* - Natural language annotation is an informal approach which appeals to the broadest audience. However, free text can be imprecise, ambiguous and its management remains solely a human responsibility.
  - *Structured* - A more formal and controlled notation such as OCL is employed to reduce ambiguity through more precise information capture. This also affords computational support on its management. Such approaches are precise but criticised as time consuming, error prone and not accessible to novice or semi skilled users.
- *Core Language Extension* - Where a finite set of common constraints is identified (often recurring patterns) and available for use as specialised predefined model constructs.

## Business Rules Approach

The business rule-oriented IS development approach is becoming more widespread (Kardasisa and Loucopoulos 2004). The business rules approach focuses on the articulation of guidelines and restrictions on states and processes which are ultimately realised in the IS system. Business Rule Management Systems usually support a repository with functionality to store, validate, retrieve, represent and manipulate rules. Business rules are especially useful in compliance intensive business domains such as the finance and insurance sectors. The various approaches seek to formalise critical business rules in a language both the manager and technologist understand. Notations usually capture the following information about each rule:

- *Name* - should give you a good idea about the topic of the business rule.



- *Specification* - rule body which is frequently based on ECA paradigm (ON (circumstance when relevant) IF (control condition), DO (action taken)).
- *Source* - who created and is responsible for the rule.
- *Related rules* - is provided to support traceability between rules.
- *Enforcement status* - e.g. active, abandoned.
- *Enforcement level* - strength of obligation e.g. preferred, mandatory.

For concrete examples see (Kardasisa and Loucopoulos 2004) and (Bajec and Krisper 2005).

## 2.7.2 Process-Oriented Models for Process Modelling

Business Process Design is centred on the capture of existing business processes for various purposes (e.g. for understanding, for documentation, for communication, for Business Process Reengineering efforts involving the analysis, simulation and improvement of processes). A multitude of languages exist for process capture perhaps the most prominent being BPMN (Owen and Raj 2003). Two themes are of particular relevance to this study. First, the emergence of Web based modelling tools and second, the emergence of hybrid modelling languages.

### **Hybrid and Declarative Languages**

Recently it has become important for process modelling methods to accommodate the articulation of process-related rules and regulations. Process model enrichment seeks to enhance enterprise models (business processes) with compliance requirements. The work presented in (Sadiq, Governatori et al. 2007) propose an approach based on control tags to visualise internal controls on process models. A traditional procedural model is augmented by addition of control objectives modelled in Formal Contract Language. The control objectives cover the task, resource and data aspects. Related work has integrated notions of risk on EPCs (zur Meuhlen and Rosemann 2005). (Goedertier, Haesen et al. 2007) is a fully declarative approach for process modelling.

### **Web-based Process Modelling Tools**

Lombardi Blueprint (IBM 2010) is a recent Web-based process documentation and repository tool. Lombardi Blueprint assists domain experts who do not necessarily have modelling expertise to create and analyse high-level diagrams in BPMN. The Web-based

‘on-demand’ solution is realised using recent Web technologies of Ajax (Garrett 2005) and the Google Web Toolkit (Johnson 2006) which provide a rich user experience.

Oryx (Oryx 2010) is a web-based process model editor, again based on the BPMN language. Collaboration is a key concern of the tool. Models can be shared with other people in a team. However, synchronous collaboration on models themselves is not facilitated. The tool is also readily extensible as it is simple to add new modelling languages to the system.

The GoogleWave (Google 2010) platform as described in Section 2.5.1 is extensible through ‘gadgets’ which are applications which plug into the framework to leverage its web presence and group working facility. ProcessWave (ProcessWave 2010) and Gravity (SAP 2010) focus on collaborative business process modelling with traditional imperative languages for with standard modelling techniques such as UML, EPC and BPMN.

### 2.7.3 Discussion

The system-oriented and process-oriented methods provide their own representational notations for constructing a set of models during the development life cycle for a given system.

UML is focused on object behaviour and is primarily devoted to supporting the software development process, from architecture design to implementation and is conceived for use by technically skilled developers. Whilst notations such as OCL may not be directly transferable, to BPM, work in the field of IS development does further support the viability of combining procedural and declarative information. The Business Rules approach and hybrid process modelling languages highlights the importance and applicability of declarative information. Finally the trend to towards Web-based deployments has also been highlighted.

## 2.8 Overall Conclusions

Four issues have emerged which inhibit the ability the ability of the above surveyed techniques and technologies to effectively provide flexible process support to EKP. The following critical deficits represent the current challenges for the flexible support of EKP:

### **The Balance of Support**

The business environment is highly dynamic. Flexibility refers to the degree to which a process can adapt to changing or unknown domain requirements (Weber, Sadiq et al. 2009). The ability to quickly create and adapt work plans is fundamental due to the unpredictability of requirements of the work. Plans remain contingent on decisions, data, events and circumstances that will not manifest themselves until closer to and during runtime (even if the final decision is based purely on preference rather than any business efficiency or regulation reasons). The PAIS should then support users to make contextualised decisions about how to perform business processes through the individualism of instance models. On the other hand, flexibility cannot come at the price of process control. Control objectives stemming from regulations and standards are becoming increasingly important for businesses; this is known as process compliance (Sadiq, Governatori et al. 2007). Organisations that understand this balance can prepare so control can be maintained over the changing situations. Providing a workable balance between flexibility and control is an enduring organisational challenge (Sadiq 2005).

Many current approaches lack support for emergent process structure and flexible run-time adaptation as the balance of maintaining process structure integrity whilst catering for individual needs is not mastered. Often processes are brittle rather than flexible. Where the issue has been addressed, systems frequently employ highly formal, mathematical languages requiring the involvement of a technical expert to handle unforeseen events.

### **Collaboration**

Due to globalisation, the planning and performance of EKP is increasingly performed by agile distributed networked organisations. Agile organisations are characterised by distributed teams consisting of team members from different partner organisations working collaboratively on joint projects. Many processes have multiple stakeholders and participants, as shown in Table 1-3. Increasing numbers of organisations are adopting agile forms and the prediction that use is expected to continue to grow (Powell, Piccoli et al. 2004). Given that group working permeates throughout process related activities, it is surprising how generally groupware and PAIS have remained distinct. PAIS have not fully benefited from potential of an incorporation of the ideas of CSCW. There is inadequate collaboration and coordination support for the joint creation, analysis, execution and adaptation of plans and process structures.

Process-centric collaborative work was identified as an important challenge for the future of the discipline a long time ago (Miers and Hunt 1995), but still remains an open challenge.

### **Accessibility**

The form of many agile organisations is temporary and fluid. For example, frequently, smaller organisations will collaborate to pool resources to provide a coherent service to compete with larger organisations. There is still a great need for temporary integration across organisational boundaries (Collins, Ketter et al. 2010). Dominant desktop applications can be cumbersome to deploy (for administrators) and consume (for users) which discourages take up and slows down inter-organisational connections.

Constantly evolving organisational forms require software solutions to be rapidly deployable to encourage the rapid connection of organisations and people to process work. Furthermore, ease of participation for individuals and organisations is necessary. Exploiting forms of distributed businesses to leverage resources regardless of location has become an essential competence. A software solution is required that can facilitate process enactment across enterprise boundaries through quick connection and disconnection to process work. Deployment must be rapid to take advantage of opportunities and services.

### **Integrated Solutions**

Current information technology for flexible process support is fragmented. Tools are often developed in separate environments using varying technologies. We have seen the specialism of the majority of support systems. Support is mostly fixed and resides towards the extremes of spectrum. We have also seen how a process can vary with respect to the most appropriate support. Integrated systems which can span a wider area of the specificity spectrum are required. Process support options should be varied to the same extent as processes themselves.

### **Overview**

Table 2-2 gives a summary of the problems when current PAIS are applied to EKP which have been extracted through analysis of related work.

Table 2-2 – Existing Problems Overview

| <b>Problem</b>  | <b>Cause</b>   | <b>Impact</b>  |
|---|--|--|
| Providing a workable balance between process flexibility and control.   | Predefined process models do not facilitate the required level of process individualism.                   | Inability for knowledge workers to respond to particular process conditions.   |
|   | Compliance requirements are not explicitly managed.  | Semantic errors in process models are possible.  |
|   | Where balance has been addressed, use of formal mathematical languages.                                    | Barrier to usage is heightened as strong expertise is required.  |
| Inadequate collaboration and coordination support for the joint creation, analysis, execution and adaptation of process structures. | No tightly integrated support for synchronous group working with rich user experience.                     | Social quality (agreement between stakeholders) is difficult slower to achieve.  |
| Slow deployment and connection to process environment for organisations and individuals.  | Traditional IS deployment paradigm (distribute-install-configure) still dominates.                         | Organisations lack agility as the rapid formation of teams is hindered.<br>Individuals are restricted by cumbersome access to process environment. |
| Disintegrated tool support.   | Highly targeted and specialised tools which are often incompatible and delivered in separate environments. | Complex often incoherent process environment.<br>Difficult development of process environment due to a multitude of underlying technologies.       |

Each deficit above is critical; failing to address one of them will seriously limit the applicability of a system and discourage take-up of that system. Whilst some of the systems surveyed address one or two of the critical areas identified above, none address all. An integrated solution is required.

## 2.9 Summary

This chapter has provided a broad perspective on the current state of PAIS. The construction of PAIS (incorporating appropriate process representation and interaction mechanisms) is a typical meta-design problem (Fischer, Giaccardi et al. 2004). The sheer number of alternative approaches indicates that it is a very difficult problem and one worthy of continued research effort. Potential techniques and tools have been examined in light of their possible application to EKP. Several weaknesses have emerged. These weaknesses provide motivation for the remainder of the study. In particular, the rise of interest in declarative languages in BPM justifies further empirical investigations into their presumed advantages over more traditional, imperative alternatives. How the Web can be better leveraged in the field is also very important.

# Chapter 3

---

## An Integrated Framework for Flexible Process Support

The chapter opens with a summary of requirements for a PAIS which can support EKP. Section 3.2 gives an overview of an integrated framework which can be applied for the flexible support of EKP. The framework consists of broad PAIS design dimensions which break down into a detailed conceptualisation which is realised with an implementation strategy. The remainder of the chapter describes the conceptualisation component of that framework.

The conceptualisation component of the framework is made up of choices made in two key dimensions of a PAIS; how to address process-orientation and how to address collaboration. The chapter is organised around detailing the choices made for these aspects and why they were made. Section 3.3 clarifies the interactive modelling paradigm for process support. A hybrid modelling scheme used to underpin an interactive modelling approach is presented in Section 3.4. An important part of the approach is based on the integration of collaborative working opportunities with process-oriented environments. How this union can be achieved is described in Section 3.5. Section 3.6 extends the conceptualisation of the interactive modelling approach by examining the impact of a hybrid modelling language and group working on how the interactive models can be managed on day-to-day basis. Throughout the chapter, a running scenario is built-up to when appropriate to demonstrate the concepts which comprise the framework.

### 3.1 High Level Requirements

The elusiveness of the problem domain addressed has already been established. A ‘wicked’ problem cannot be wholly formulated or captured. To develop a better understanding of the problem and the approaches which best can serve it, as a community, we can only identify an issue and tentatively address that issue. In this way, our knowledge of the problem domain moves forward gradually but systematically.

Chapters 1 and 2 identified the problem domain and deficits in the ability of current techniques to support EKP. Two deficits which are addressed by this conceptualisation are

given along with a description of requirements derived from the deficit. The remaining two deficits are addressed by the implementation effort, described in Chapter 4. It is important to note that not all low level requirements for a PAIS are enumerated; only those which are of particular concern to this research effort.

### **1/Balance of support**

A key requirement for EKP is to establish a PAIS which can balance:

- *Flexibility/Individualism* - The system must provide a level of manoeuvrability in instance model design which allows users to plan work as is fit for the particular needs of the *local context and* situation at hand.
- *Control/Compliance* - The system must ensure the design of each instance model conforms to any applicable organisational standards.

The mechanism/s used for this purpose must be usable by non-modelling experts.

### **2/Collaboration**

It has been shown that group working in process planning, analysis and adaption is becoming essential. Multi-stakeholder processes are growing due to the often inter-organisational nature of the modern enterprise. Groups maybe formed from users from across a company or companies, not only in terms of geography but also in terms of hierarchy.

The system must support different process participants, potentially from different partner organisations, to perform day-day indirect work (process articulation, reviewing, leaning, analysis and adaptation) jointly (through each collaboration mode (see Figure 2-3)).

## **3.2 An Integrated Framework**

Chapter 3 and 4 develop a prescriptive framework to address the deficits raised in the previous chapter. The approach can be described as a top-down development. Top-down development is where business needs demand innovation (problems derived from the problem domain were identified; a conceptual design and eventually a technical implementation were developed to address those problems). The alternate approach of a bottom-up way of working is where innovation informs and drives business practise.

The following points briefly introduce two key design dimensions of a PAIS to support EKP:

- *Process-orientation* – Two key aspects of process-orientation are how models will be deployed by the PAIS and what representation scheme will be used (users encode domain structure in models using the prescribed modelling language). This dimension will address requirement 1.
- *Collaboration* – How will ensembles be supported in the consumption of the core process-oriented services? This dimension will address requirement 2.

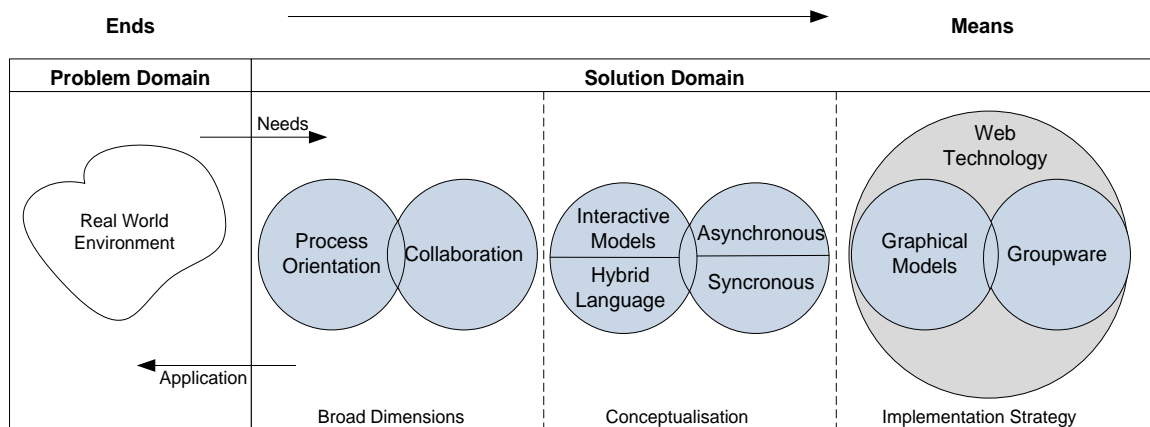


Figure 3-1 - A High Level View of the Integrated Framework

For each dimension, a choice is made regarding its treatment. This choice is driven by requirements found in the problem domain. We strive for alignment between the problem domain and the solution domain. It is the combination of treatments which make up the framework for the flexible support of EKP.

- *Process-orientation* -
  - Deployment Method - An extended interactive modelling approach is conceived. This comprises of an underlying architecture given in Section 3.3 and the impact of a hybrid modelling language and group working on interactive model management given in Section 3.6.
  - Modelling Language - A hybrid lightweight intuitive modelling language is conceived to underpin the interactive modelling approach. Language specification is presented in Section 3.4.
- *Collaboration* - a strategy for synchronous and asynchronous collaborative working opportunities is detailed in Section 3.5.



Previous approaches neglect the integration of process-orientation and rich collaborative opportunities. The goal of our approach is the tight integration of process-oriented services (i.e. work planning and work performance) and collaboration in order to coordinate the day-to-day work of distributed teams. This integration is visualised as the overlap in each of the concentric circles in Figure 3-1 and is the focus of this research.

The discussion in this chapter focuses on the conceptualisation element of the integrated framework. To complete the specification of the integrated framework for EKP, the lower level implementation detail will be described in Chapter 4. As can be seen in Figure 3.1, process-oriented services are achieved through graphical models, collaboration is achieved through groupware and the PAIS will manage and deliver content using Web technology. The conceptualisation is realised and delivered as a working academic software prototype.

Each component of the framework must be validated to ensure its fitness for purpose. The implementation effort described in Chapter 4 serves as a proof of concept. The usefulness of concept and implementation is demonstrated with an ongoing detailed scenario throughout Chapters 3 and 4. The usability of the concept and implementation is demonstrated through a usability study described in Chapter 5.

### **Scenario: Patient Rehabilitation**

To help demonstrate the various concepts which make up the approach summarised above, it will be shown how they can be applied to a typical example of EKP. A healthcare scenario has been selected.

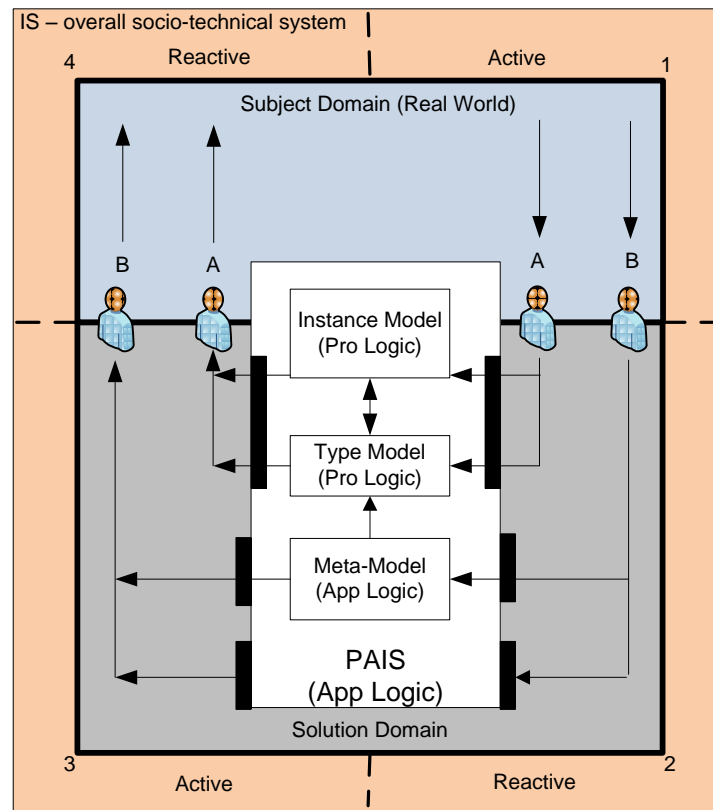
After the immediate emergency action (e.g. surgery) has been performed, the usual care time for a broken Tibia or Fibula is 12 to 16 weeks. An orthopaedic doctor will make an assessment. This bone specialist will plan further follow-up appointments, assessments and rehabilitation sessions as necessary. Other medical professionals (e.g. trainee bone specialists) and hospital staff (e.g. porters, clerical staff) will participate in the rehabilitation process. Very frequently, rehabilitation will involve a Chartered Physiotherapist to restore the range of ankle and knee movement and to restore the muscle strength that is lost during the immobilisation period. The scenario focuses on the long period of rehabilitation and the various tasks it might include rather than the initial emergency medical treatment. The scenario will focus on the task aspect.

It can immediately be seen that the scenario possesses many properties of an EKP. Firstly, rehabilitation is a complex human activity; rehabilitation relies on the skill and knowledge of medical professionals. Secondly, in the scenario above it is of paramount importance that the individual and emergent needs of the patient are addressed whilst at the same time medical professionals are bound by strict rules to ensure the quality of service given to patients. This is a clear example of the important balance between process flexibility and control. Lastly it can be seen that the rehabilitation of a patient is a team endeavour (often distributed geographically and organisationally). Each member of the team is keen to ensure the most suitable rehabilitation plan and often working together to achieve it.

### 3.3 Interactive Modelling Approach to Process Support

As established in Chapter 2, the interactive modelling paradigm is a promising approach for achieving the flexible support of EKP. The ultimate goal of the interactive modelling approach is to guide groups of knowledge workers through their day-to-day professional working activities. The purpose of this section is to further its conceptualisation to help meet the needs of EKP. An extended architecture is presented in Figure 3-2 which clarifies and enhances the interactive modelling paradigm by detailing the role of various PAIS components amongst a wider socio-technical system.

The overall IS comprises of two interrelated systems; the subject domain (the real world process) and the solution domain (the PAIS which seeks to coordinate the real world process). The basic assumption is that due to the modern enterprise environment, each system is fluid and evolving. Both of these systems should be in alignment at all times; change in one system should be reflected in the other. We can say that both systems are active and reactive. As shown in Figure 3-2, it is both the knowledge workers and the IT professional's responsibility to maintain this alignment.



A – Knowledge Worker

B – IT Professional

■ - Model/System Interactors

Figure 3-2 - The Hierarchy of Interactive Models

The PAIS establishes two types of logic:

- *Application Logic* - Established by IT professionals to provide generic and reusable process-oriented software services. A way of working is established and encoded in low level code. In terms of the rehabilitation scenario, application logic is the process-aware platform which IT professionals develop for medical professionals to use e.g. modelling tools, model analysis tools, data persistency services.
- *Process Logic* - Established by knowledge workers working with a graphical front-end (the individual process models). The PAIS is consumed to plan and perform actual work. In terms of the rehabilitation scenario, process logic is the artefact created (concrete plans of rehabilitation for each patient) by medical professional's using the PAIS.

### 3.3.1 Interactive Models for Process Logic

The focus of this work is on how knowledge workers develop process logic with interactive instance and type models to manage day-to-day work. An interactive model is a shared (the model serves as a hub), visual externalisation of process-centric information that can be created, viewed, traversed, analysed, simulated, adapted, improved and executed by process participants. The direct tangibility and visibility of the models facilitate the day-to-day model interactions required to plan, coordinate, learn, manage and perform knowledge-based work endeavours.

Interactive type and instance models sit between the knowledge workers and the PAIS helping relate the real world process to the supporting IT tools and infrastructure so they can be leveraged by that real world process. In effect, knowledge workers program a PAIS with interactive models to govern the behaviour of that PAIS. Knowledge workers work with interactive models usually through model interactors such as editing and analysis tools.

### 3.3.2 Interactive Model-Driven PAIS - Conceptual Walkthrough

The conceptualisation of Section 3.2.1 provides a way of viewing and managing the complexity of the problem domain as four zones of interest emerge. This section will step through the zones identified in Figure 3-2 with particular focus on behaviour of type and instance models to further describe the interactive modelling approach. Where appropriate, the description of each zone will draw upon the different types of work (A, B and C) (Engelbart 1992) described in Section 2.2.1 to show where they fit with the knowledge workers perspective on the interactive modelling approach.

#### **Zone 1: Active Subject Environment (Upper Right)**

The actor perceives (interprets/internalises) some phenomena in the real world process environment. Domain phenomena have influence on model composition and can come from many sources e.g. formal documentation (such as applicable legislation) and specific work needs. In terms of the healthcare scenario work needs are the particular requirements of a patient throughout their rehabilitation e.g. the specifics of the injury would be collected by the medical professional or the current condition of the patient. Eliciting relevant process information from the domain is 'B' type work.

## Zone 2: Reactive PAIS (Lower Right)

The actor compares the current state of the subject environment to the current state of the relevant PAIS component (instance, type, meta-model, PAIS). If required, the actor can make a change (articulation/externalisation) to the relevant component to bring the system into alignment. By altering a component, users can adapt the system to fit local needs and changes in the environment, thus tailoring system behaviour. Over time, each component then develops in parallel with real world knowledge and requirements as they are understood and emerge. Different model interactors are used make changes to the relevant component (e.g. an Integrated Development Environment (IDE) for the PAIS and a graphical model editor for the instance model).

Each component has forces which act upon it. If alignment, necessary for the usefulness of the system, is to be maintained, the component should respond to these forces appropriately. Table 3-1 gives a summary of the frequency of change, the actor who reacts to change and through what medium for each component.

Table 3-1 – Description of Models and their Properties

| Component         | Frequency of change | Scope of change | Actor making change | Medium                              |
|-------------------|---------------------|-----------------|---------------------|-------------------------------------|
| <i>Instance</i>   | Frequent            | Minor           | Knowledge worker    | Graphical models                    |
| <i>Type</i>       |                     |                 | Knowledge worker    | Graphical models                    |
| <i>Meta-model</i> |                     |                 | IT professional     | Graphical models and low level code |
| <i>PAIS</i>       | Less frequent       | Major           | IT professional     | Low level code                      |

For any component of the PAIS to ‘drift’ along a trajectory with no underpinning methodology risks directionless and confused development. A framework which applies a searching, gradual methodology to each component of the PAIS is required. Some examples of articulation at the higher levels are given (‘C’ type work):

- *PAIS* - The improvement of the PAIS with system level enhancements e.g. increased security, performance or compatibility.
- *Meta-model* - The extension of the current meta-model (language) used for type and instance model articulation e.g. new concepts are introduced.

The focus here is on the instance and type models which capture common understanding in terms of how work will proceed. The needs of particular day-to-day working conditions are reflected in the instance model to help coordinate work. Users are empowered to cope with

changing knowledge of their work. This is 'B' type work; participants plan work through type and instance models until are satisfied that the model achieves work needs.

Stimulus to model change:

- *Internal* - Need for change is generated through model inspection and model visualisation/analysis tools which help identify problems.
  - *Type* – For example, a type model for the handling of a broken tibia is inspected by a group of medical professionals before it is deployed to highlight opportunities for optimisation and improvement.
  - *Instance* – For example, computational supports are applied to an instance model (the work plan for a particular patient) to highlight opportunities for optimisation and improvement. The verification view (see Section 3.4.5) is an example of such a computational support which determines the correctness of the instance model with respect to any defined constraints.
- *External* - Need for change is generated through new information emerging in the real world process. Change is required to maintain synchronisation with the subject environment.
  - *Type* - A change in applicable legislation/standards may result in new working practices being incorporated into a type model. For example, a revised code of practise may state that the same orthopaedic Doctor must perform the initial assessment and final assessment of a patient.
  - *Instance* - The emergent needs of a patient would need to be reflected in the design of the instance model. For example, the patient may not respond to treatment as expected which causes the remainder of the plan being adjusted.

Types of model change:

- *Model composition* - Change of model state is made through explicit modification e.g. on-the-fly work (re)design. Structure can emerge during enactment as participants make decisions regarding ambiguous parts of the model (specification has been deferred and the composition of multiple tasks and resources is made when known) or alter the model to reflect improved understanding. Each of the examples given above would result in a change in model composition.
- *Model state* - Change of model state is made through model enactment. It is not the composition of the model which changes but its execution state. Such execution

occurs at the instance level and involves model state changes such as a task changing from state 'ongoing' to state 'completed'.

It is important that new information from the real world environment is reacted to within the PAIS. It is the flexibility of interactive type and instance models which helps the PAIS to accommodate environmental change and local contexts (i.e. stay in alignment with real world needs through the medium of the graphical interactive model).

Response outside of the PAIS is problematic as it would lead to a lack of governance, a lack of reasoning about work design and a state of confusion amongst actors as outdated models would persist in the work environment. For example, if a medical professional performed a treatment but that treatment was not reflected in the plan, then other people using the plan would not be aware that the treatment occurred. This could easily lead to serious problems e.g. a conflicting treatment could be planned for the patient. In addition post-work analysis of the case would be uninformative as the plan was not a true record of what occurred.

### **Zone 3: Active PAIS (Lower Left)**

The actor perceives (interprets/internalises) information from the components of the PAIS. At the instance levels, a graphical model serves as the process awareness mechanism for the knowledge workers; a shared coordination mechanism. People inspect the model to see what they should do and when they should do it, in a wider context of what other people have done and will do. Knowledge workers can visualise and navigate the entire process definition making forward planning and retrospection easier through visibility of contextual information (past and planned activities in process context, documents related to this process, other actors involved etc.) For example a medical professional would look at the instance model to understand the current state of the case of a patient before deciding on what should be planned or performed next. Model interpretation is a form of 'B' type work.

### **Zone 4: Reactive Subject Environment (Upper Left)**

At the instance level, equipped with the knowledge gained from model interpretation, actors perform some action to drive the real world process forward:

- *Manual* - A knowledge worker performs some work. This may employ an external productivity tool such as creating a document with MS word or be a physical task such as examining a patient.

- *Automated* - The tasks which comprise the process can be performed computationally. For example a task in a process may be fully automated such as sending out an invoice for patient treatment or a payroll system for hospital staff.

‘A’ type work occupies zone 4; the instance model is enacted and work is performed.

## 3.4 A Hybrid Modelling Language for the Interactive Modelling Approach

The second part of conceptualisation is the modelling language used to underpin the interactive modelling approach described in Section 3.3. What language should be used to meet the requirements identified in Section 3.1? In this section we will describe how a hybrid modelling language can contribute to achieving these requirements. A brief introduction to language use is followed by the specification of the language itself.

### 3.4.1 The Hybrid Modelling Approach

The approach is based on the concept of ‘pockets of flexibility’ (Sadiq, Orlowska et al. 2005). For this consideration a ‘pocket’ is taken to mean a single page of process description where the total process description can be made up of multiple pockets organised in a work breakdown structure (hierarchical). Tasks form part of process description and are used to arrange work hierarchically. Tasks can be composite (a complex task) or atomic (simple). A composite task is made up of sub-tasks to enable work to be arranged vertically as well as horizontally. This is seen in Figure 3-3 where tasks ‘A’ and ‘C’ are composite tasks and the remainder are atomic (as no sub-structure is defined).

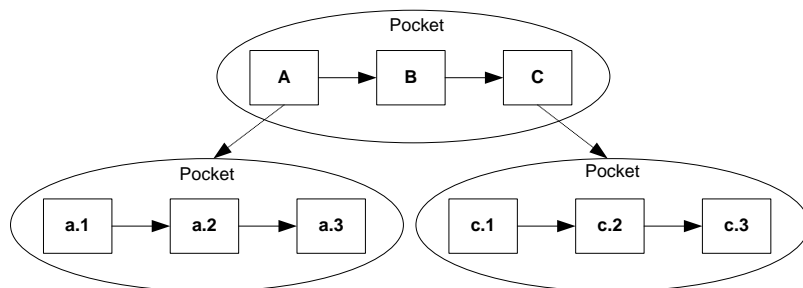


Figure 3-3 - Pockets of Flexibility

Interspersed throughout the entire process description are ‘pockets’ of space where process definition structure can be composed before and during enactment. The organisation of a



single pocket is first described. Section 3.5.5 describes how the approach can be used with hierarchical process structures.

### **‘Pocket’ Content**

As highlighted in Chapter 2, within the BPM community, there is a great interest in the combination of procedural (processes) and declarative (rules) information as a way to address some of the current challenges posed by flexible process support. This certainly applies to the challenge posed by requirement 1.

With a hybrid modelling language, a single ‘pocket’ may include both procedural (P) and declarative (D) content. The two types of content are separate but co-dependent as the state of one depends on the state of the other.

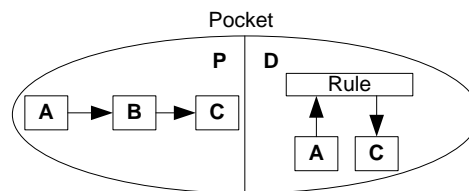


Figure 3-4 - Structure of a Single ‘Pocket’ of Work Specification

Each type of content (P and D) has a specific purpose:

- *P* - A governed procedural model is used to articulate traditional operational plans of work. *P* is often created ‘just in time’ and is subject to frequent ad-hoc modifications. *P* establishes work plans; arranging the tasks (e.g. initial assessment, interim assessments, physiotherapy sessions etc), their scheduling and the people (surgeons, physiotherapists etc) who have been assigned to perform them for each patient.
- *D* - A governing declarative model is used to articulate constraints (properties which should hold in the design of any *P* which resolves *D*). *D* is not used to prescribe precisely how participants perform their work. *D* is used to guide the design of *P* by capturing the boundaries of design possibilities/properties to which it should adhere. *D* is generally more stable and known in advance. In terms of the healthcare scenario, content in *D* establishes the reusable rules which would apply to the handling of all patients such as ‘if the task ‘discharge patient’ is performed then the task ‘prepare patient for discharge’ should have preceded it.’

In essence, a partial process specification is made in advance by D. This specification is used together with domain requirements at runtime to guide the specification of P which is the concrete work plan. P is created only when the details of the work emerge closer to the time of work performance. D forms a basis from which an objective appraisal of P's correctness can be made.

While a transformation of declarative models into well-conceivable procedural models implies overhead (a further burden on the user), the approach moves towards achieving a balance between process flexibility and control as a large, often infinite number of possible designs of P are available to users whilst it can be ensured that each of those designs conforms to the captured compliance rules in D. In other words, the user is free to model anything which falls within the space carved out by the D. Table 3-2 gives a summary of the various properties of each type of content (P and D).

Table 3-2 – Summary of Procedural and Declarative Models

| <b>Descriptor</b>                      | <b>P</b>   | <b>D</b>  |
|--|--|---|
| <i>Paradigm</i>                        | Procedural   | Declarative   |
| <i>Medium</i>                          | Graphical  | Graphical   |
| <i>Function</i>                        | Articulate executable work plan  | Articulate constraints on work plan design  |
| <i>When made</i>                       | Usually 'just in time' for work performance and ongoing throughout that work | Usually in advance of work performance  |
| <i>Domain requirements</i>             | Local and contextualised conditions of work                                  | Regulatory bodies, codes of practice and business partner contracts project specific rules. |
| <i>Frequency of update</i>             | Frequent   | Infrequent  |
| <i>Who creates/ updates</i>            | Knowledge worker   | Knowledge worker  |
| <i>Who consumes</i>                    | Knowledge worker   | Knowledge workers   |
| <i>Governing forces on composition</i> | Meta-model (of P)<br>Domain requirements<br>D                                | Meta-model (of D)<br>Domain requirements  |

### **Governing Forces on Model Composition**

There are two main forces on the form of both P and D:

- *Meta-model* - Both P and D must adhere to their relative meta-models (described in Sections 3.4.2 and 3.4.3). The meta-model can be built into the model editor of the PAIS.
- *Domain Requirements* - The approach relies heavily on user's knowledge of the domain. The user must build both P and D to reflect domain requirements to ensure semantic quality of each model. Through model inspection the knowledge worker must ensure its completeness (all domain information is reflected) and correctness (with respect to business requirements). Two correctness aspects for D can be

identified; validity (stated in such a way that always a solution (free from conflict)) and minimality (stated in such a way that excludes redundancy).

D and P can be related to each other. P should satisfy the constraints established by D. One procedural design should emerge from all possible designs such that no constraint in D is violated by P. Therefore, P has an extra governing force:

- *D* - The user must build P to reflect the constraints captured in D.

We can say D establishes a basis for an objective check on the semantic quality of P. However, as much as semantic correctness of models can be facilitated through linguistic mechanisms, it is still dependant on the knowledge workers and their understanding of domain requirements. The approach can't guarantee the quality of P or D with respect to domain requirements as this is driven by the knowledge workers:

- *D* - D may not represent domain requirements as it could be incomplete (e.g. it does not reflect a particular healthcare regulation) or it could be inaccurate (e.g. a healthcare regulation has been misinterpreted and not captured correctly by D).
- *P* - P may be valid with respect to D but the P does not meet domain requirements as it could be incomplete (e.g. the full rehabilitation programme has not yet been planned) or it could be incorrect (e.g. the wrong treatment has been planned for the actual needs of the patient).

The general way of working has now been established. The difference between the approach presented here and the original pockets of flexibility approach is the deployment paradigm it is combined with (interactive models rather than workflow) and the language used (an extended declarative modelling language is presented in the next section). The significance of an interactive modelling approach is the extension of the role of work performer to include work designer and the facilitation of direct interaction with declarative models.

The challenge is to bring this way of working into the day-to-day working environments of users in such a way as not to violate the original vision of accessible interactive models whilst maintaining a convergence of model paradigms which is systematic and well structured.

Each type of content has a meta-model. Meta-models establish categories of useful concepts and links between concepts in the subject domain. Furthermore, the meta-model establishes rules which restrict how concepts can be related through links. The following sections will describe the languages of each type of content in more detail.

### 3.4.2 Declarative Models

It is important that the constraint model should allow for greater or less devolved decision-making in process authoring. In other words, D could be empty, loose (many designs of P satisfy D) or tight (few designs of P satisfy D). It is vital to understand that the more constraints D has, the less freedom users will have for composing P. The less constraints D has, the more freedom users will have for composing P.

The declarative language presented is a small set of patterns useful in process-oriented domains. It is a domain-oriented approach. The language then serves as a test-bed to examine the concept of hybrid models introduced in the previous section. Should the concept with this initial test-bed language prove promising, the language can be adapted for applicability in different domains.

Although the language is small, considerable scope for expressivity is still maintained. The intentional, specific, predefined language implemented will afford articulation of static task and resource aspect constraints. The rationale behind this strategy can be found in Appendix A.

#### **Language Concepts**

*Task Types* - Multiple instances of a task of the same task type are often required in a process. D establishes a set of user defined task types (T). The collection of task types defined in D forms a predefined pool of tasks which may or may not be used in the definition of P. Once a task type is defined once in D, it can be instantiated multiples times in the formulation of P.

Part of the definition of a task type restricts the number of times a task type in D can be instantiated in P. Each task type in D has attributes specifying the minimum and maximum number of times it can be instantiated. To continue the healthcare scenario, the following task types can be identified:

- Cast (0 - n) - A cast is applied to the patients arm.
- Sling (0 - n) - A sling is applied to the patients arm.
- Medicate (0 - n) - Medication is prescribed to the patient.
- Assess (1 - 2) - Post surgery, an examination is performed to assess the needs of rehabilitation.
- Interim Assessment (1 - n) - A progress check on the condition of the patient various decisions are made here e.g. whether to remove the cast, weight bearing checks or whether to locate the patient at home.
- Remove Cast (0 - n) - The cast is removed from the patient.
- Plan Exit (0 - n) - Hospital staff arrange how the patient will be transferred home
- Practical (0 - n) - A medical professional will take the patient through practical day to issues such as diet, how to relieve swelling and infections, appropriate clothing and sanitation guidance.
- Physio Session (0 - n) - The physiotherapist performs various exercises with the patient; hydrotherapy exercises, manual therapy.
- Exit (0 - n) - The patient is transferred to nominated accommodation.

To give an example of the type-instance paradigm at the object level (components of a process), a ‘physio session’ has been declared as a task type in D. The ‘physio session’ task type can now be instantiated as many times as required in the composition of P i.e. multiple ‘physio sessions’ maybe required during the rehabilitation process.

*Constraint Types* - Task types serve as parameters (in either a trigger (Tr) or target (Ta) role) to constraint types to make constraint expressions. There are a set of 7 predefined constraint types described in tables 3-3 to 3-9. For the following consideration:

- Tr is a nonempty set of task types.
- Ta is a nonempty set of task types.
- In1 is the set of task instances present in P whose task type appears in set Tr.
- In2 is the set of task instances present in P whose task type belongs to the set Ta.
- m is the number of instances in set In1.

Table 3-3 – The Serial Constraint

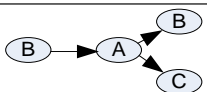
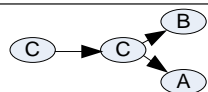
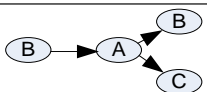
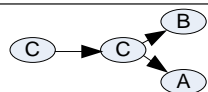
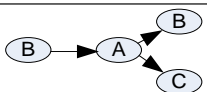
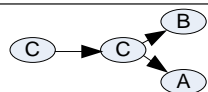
| Name                | Serial (Non-conditional composition constraint)   |   |             |           |               |  |   |   |  |
|---------------------|---|---|-------------|-----------|---------------|--|---|---|--|
| Specification       | <p>Parameters: (Ta)</p> <p>Attributes: Name</p> <p>In P, there should exist a path (either exclusively back or forward) from each member of In2 to every other member of In2. It is not implied that instances of each member of Ta have to be found in In2. Consecutive placement of tasks in In2 is not implied. No particular ordering of tasks (beyond serial placement) in In2 is implied.</p> |   |             |           |               |  |   |   |  |
| Further Description | <table><tr><th>Constraint</th><th>Well formed</th><th>Malformed</th></tr><tr><td>Serial (A, B)</td><td></td><td></td></tr></table>   | Constraint  | Well formed | Malformed | Serial (A, B) |  |  | <p>The model on the right is malformed as there is no path from A to B.</p> |  |
| Constraint          | Well formed   | Malformed   |             |           |               |  |   |   |  |
| Serial (A, B)       |   |  |             |           |               |  |   |   |  |
| Example             | <p>Serial (Pysio Session)</p> <p>This asserts that each ‘physio session’ task instance found in P should be performed in serial order (i.e. one at a time). This is used as the result of one ‘physio session’ is used to guide the activities of the next and cannot be performed in parallel.</p>   |   |             |           |               |  |   |   |  |

Table 3-4 – The Fork Constraint

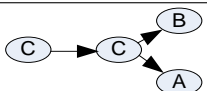
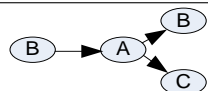
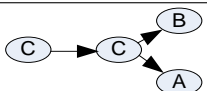
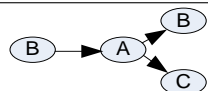
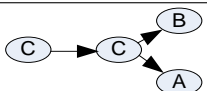
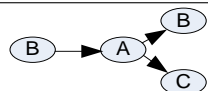
| <i>Name</i>                | <b>Fork (Non-conditional composition constraint)</b>   |   |             |           |             |  |   |  |  |
|----------------------------|--|---|-------------|-----------|-------------|--|---|--|--|
| <i>Specification</i>       | <p>Parameters: (Ta)</p> <p>Attributes: Name</p> <p>In P, there should not exist a path (either exclusively back or forward) from any member of In2 to any other member of In2. It is not implied that instances of all members of Ta have to be found in In2. Consecutive placement of tasks in In2 is not implied. No particular ordering of tasks (beyond fork placement) in In2 is implied.</p> |   |             |           |             |  |   |  |  |
| <i>Further Description</i> | <table><tr><th>Constraint</th><th>Well formed</th><th>Malformed</th></tr><tr><td>Fork (A, B)</td><td></td><td></td></tr></table>  | Constraint  | Well formed | Malformed | Fork (A, B) |  |  | <p>The model on the right is malformed as there is a path from A to B.</p> |  |
| Constraint                 | Well formed  | Malformed   |             |           |             |  |   |  |  |
| Fork (A, B)                |    |  |             |           |             |  |   |  |  |
| <i>Example</i>             | <p>Fork (Assess)</p> <p>This asserts that each ‘assess’ task instance found in P should be performed in a fork (often in parallel). This is used as the result of one assessment should not affect the other i.e. the patient is assessed independently by two separate people.</p>  |   |             |           |             |  |   |  |  |

Table 3-5 – The Exclude Constraint

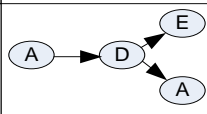
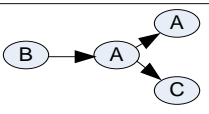
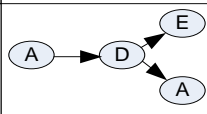
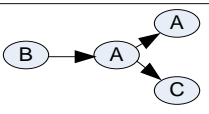
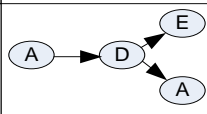
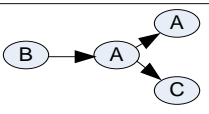
| <i>Name</i>                | <b>Exclude (Conditional composition constraint)</b>   |   |             |           |                       |  |   |
|----------------------------|---|---|-------------|-----------|-----------------------|--|---|
| <i>Specification</i>       | <p>Parameters: (Tr, Ta)</p> <p>Attributes: Name, Target Mode (Any, One, All)</p> <p>If m = 0 then no constraint is enforced. If m &gt; 0, then In2 should not contain:</p> <ul style="list-style-type: none"><li>Any – Any instances of any task type in Ta</li><li>One – Any instances of only one task type in Ta</li><li>All – Any instances of each task type in Ta</li></ul>   |   |             |           |                       |  |   |
| <i>Further Description</i> | <table><tr><th>Constraint</th><th>Well formed</th><th>Malformed</th></tr><tr><td>Exclude ((A), any(B))</td><td></td><td></td></tr></table> <p>The model on the right is malformed as an A task and B task is present in P.</p>   | Constraint  | Well formed | Malformed | Exclude ((A), any(B)) |  |  |
| Constraint                 | Well formed   | Malformed   |             |           |                       |  |   |
| Exclude ((A), any(B))      |   |  |             |           |                       |  |   |
| <i>Example</i>             | <p>Exclude ((sling), all(cast, remove cast))</p> <p>The exclude constraint establishes a dependency between two sets. The presence of an instance of a triggering task type in P implies the target condition holds in P. In this case, if a ‘cast’ task appears in P, then a ‘sling’ task should not appear in P. This basically establishes exclusive choice between the cast and sling tasks as both tasks should not be performed in P.</p> |   |             |           |                       |  |   |

Table 3-6 – The Include Constraint

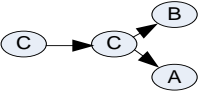
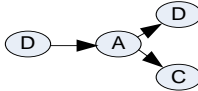
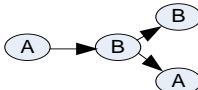
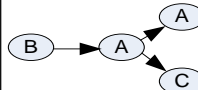
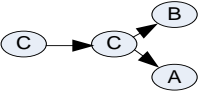
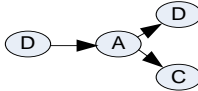
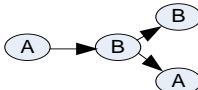
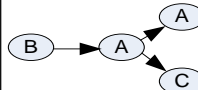
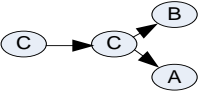
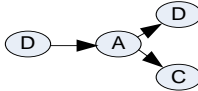
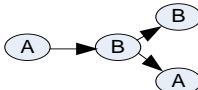
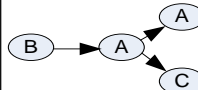
| Name                                 | Include (Conditional composition constraint)  |   |             |           |                                    |  |   |            |             |           |                                      |  |   |
|--------------------------------------|---|---|-------------|-----------|------------------------------------|--|---|------------|-------------|-----------|--------------------------------------|--|---|
| Specification                        | <p>Parameters: (Tr, Ta)</p> <p>Attributes: Name, Target Mode (Any, One, All), Constraint (Weak, Strong)</p> <p>If m = 0 then no constraint is enforced. If m &gt; 0, then In2 should contain;</p> <ul style="list-style-type: none"><li>Any – At least 1/m, for weak/strong respectively, instance/s of any task type in Ta</li><li>One – At least 1/m, for weak/strong respectively, instance/s of only one task type in Ta</li><li>All – At least 1/m, for weak/strong respectively, instance/s of each task type in Ta</li></ul>   |   |             |           |                                    |  |   |            |             |           |                                      |  |   |
| Further Description                  | <table><tr><th>Constraint</th><th>Well formed</th><th>Malformed</th></tr><tr><td>Include<br/>((C), any(B))<br/>(weak)</td><td></td><td></td></tr></table> <p>The model on the right is malformed as a C is present but B is not.</p> <table><tr><th>Constraint</th><th>Well formed</th><th>Malformed</th></tr><tr><td>Include<br/>((A), any(B))<br/>(strong)</td><td></td><td></td></tr></table> <p>The model on the right is malformed as a two A tasks are present and but only one B task.</p> | Constraint  | Well formed | Malformed | Include<br>((C), any(B))<br>(weak) |  |  | Constraint | Well formed | Malformed | Include<br>((A), any(B))<br>(strong) |  |  |
| Constraint                           | Well formed   | Malformed   |             |           |                                    |  |   |            |             |           |                                      |  |   |
| Include<br>((C), any(B))<br>(weak)   |   |  |             |           |                                    |  |   |            |             |           |                                      |  |   |
| Constraint                           | Well formed   | Malformed   |             |           |                                    |  |   |            |             |           |                                      |  |   |
| Include<br>((A), any(B))<br>(strong) |   |  |             |           |                                    |  |   |            |             |           |                                      |  |   |
| Example                              | <p>Include weak ((Medicate), all (Practical Session))</p> <p>The include constraint establishes a dependency between two sets. The presence of an instance of a triggering task type in P implies the presence of the target condition. In this case, if a task of type ‘medicate’ appears in P, then a task of type ‘practical session’ task should also appear in P. The practical session may instruct the patient of issues related to the medication. This constraint uses the weak option which means that only one ‘practical session’ task is required to cover the constraint for multiple ‘medicate’ tasks.</p>   |   |             |           |                                    |  |   |            |             |           |                                      |  |   |



Table 3-7 – The Precede Constraint

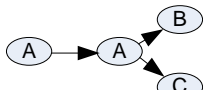
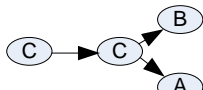
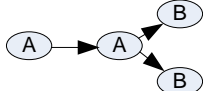
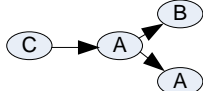
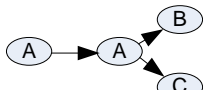
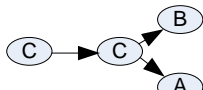
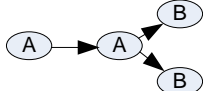
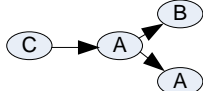
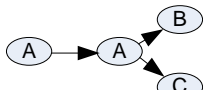
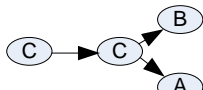
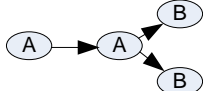
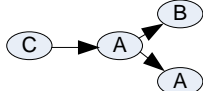
| Name                                 | Precede (Conditional composition constraint)   |   |             |           |                                    |  |   |            |             |           |                                      |  |   |
|--------------------------------------|--|---|-------------|-----------|------------------------------------|--|---|------------|-------------|-----------|--------------------------------------|--|---|
| Specification                        | <p>Parameters: (Tr, Ta)</p> <p>Attributes: Name, Target Mode (Any, One, All), Constraint (Weak, Strong)</p> <p>Each member of In1 has a set (Inx) made up of the task instances present in P which lies in the upstream for that member of In1. Each member of In1 has a set (Iny) made up of the task instances present in P which lies in the immediate upstream of that member of In1.</p> <p>If <math>m = 0</math> then no constraint is enforced. If <math>m &gt; 0</math>, then for each member of In1, Inx (or Iny if chain mode is selected), should contain;</p> <ul style="list-style-type: none"><li>Any – At least 1/m for weak/strong respectively instance/s of any task type in Ta</li><li>One – At least 1/m for weak/strong respectively instance/s of only one task type in Ta</li><li>All – At least 1/m for weak/strong respectively instance/s of each task type in Ta</li></ul>  |   |             |           |                                    |  |   |            |             |           |                                      |  |   |
| Further Description                  | <table><tr><th>Constraint</th><th>Well formed</th><th>Malformed</th></tr><tr><td>Precede<br/>((A), any(B))<br/>(weak)</td><td></td><td></td></tr></table> <p>The model on the right is malformed as an A is present but no B is present in its upstream (in this case the upstream is empty).</p> <table><tr><th>Constraint</th><th>Well formed</th><th>Malformed</th></tr><tr><td>Precede<br/>((A), any(B))<br/>(strong)</td><td></td><td></td></tr></table> <p>The model on the right is malformed as an A is present which has no B in its upstream.</p>  | Constraint  | Well formed | Malformed | Precede<br>((A), any(B))<br>(weak) |  |  | Constraint | Well formed | Malformed | Precede<br>((A), any(B))<br>(strong) |  |  |
| Constraint                           | Well formed  | Malformed   |             |           |                                    |  |   |            |             |           |                                      |  |   |
| Precede<br>((A), any(B))<br>(weak)   |    |    |             |           |                                    |  |   |            |             |           |                                      |  |   |
| Constraint                           | Well formed  | Malformed   |             |           |                                    |  |   |            |             |           |                                      |  |   |
| Precede<br>((A), any(B))<br>(strong) |    |  |             |           |                                    |  |   |            |             |           |                                      |  |   |
| Examples                             | <p>Precede weak chain ((plan exit), all (exit))</p> <p>Precede strong ((exit), all (practical))</p> <p>Precede strong ((Apply cast), any (physio session, prescribe medication))</p> <p>Precede weak ((Assess), one(cast, sling))</p> <p>The precede constraint establishes a dependency between two sets. The presence of an instance of a triggering task type in P implies the presence of the target condition in the upstream of the triggering task. In this case:</p> <ul style="list-style-type: none"><li>If a plan exit task appears in P, then an exit task should appear directly after it in P.</li><li>If an exit task appears in P, then a practical task should appear after it in P.</li><li>If an apply cast task appears in P, then both a rehabilitation session and a prescribe medication task should appear after it in P.</li><li>If an assessment is task is performed in P, then either a cast or sling task should be performed after it.</li></ul> |   |             |           |                                    |  |   |            |             |           |                                      |  |   |

Table 3-8 – The Succeed Constraint

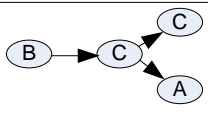
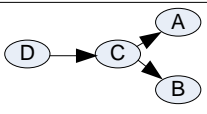
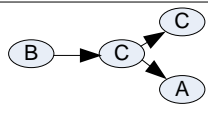
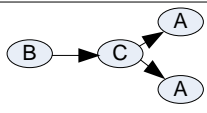
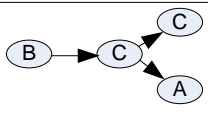
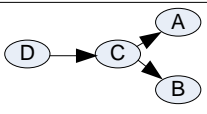
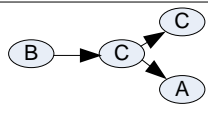
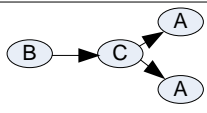
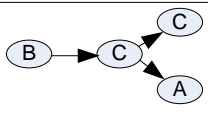
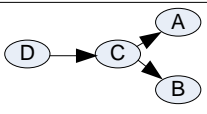
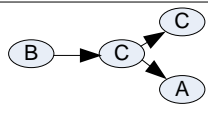
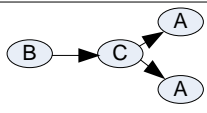
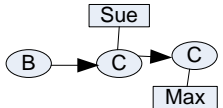
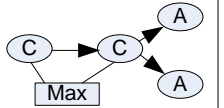
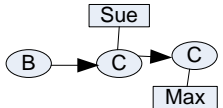
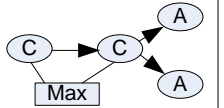
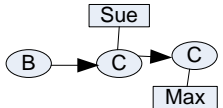
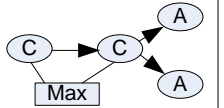
|                                      |   |   |             |           |                                    |   |  |            |            |           |                                      |  |   |
|--------------------------------------|---|---|-------------|-----------|------------------------------------|---|--|------------|------------|-----------|--------------------------------------|--|---|
| Name                                 | Succeed (Conditional composition constraint)  |   |             |           |                                    |   |  |            |            |           |                                      |  |   |
| Specification                        | <p>Parameters: (Tr, Ta)</p> <p>Attributes: Name, Target Mode (Any, One, All), Constraint (Weak, Strong)</p> <p>Each member of In1 has a set (Inx) made up of the task instances present in the P which lies in the downstream of that member of In1. Each member of In1 has a set (Iny) made up of the task instances present in P which lies in the immediate downstream of that member of In1.</p> <p>If <math>m = 0</math> then no constraint is enforced. If <math>m &gt; 0</math>, then for each member of In1, Inx (or Iny if chain mode is selected), should contain;</p> <ul style="list-style-type: none"><li>Any – At least <math>1/m</math> for weak/strong respectively instance/s of any task type in Ta</li><li>One – At least <math>1/m</math> for weak/strong respectively instance/s of only one task type in Ta</li><li>All – At least <math>1/m</math> for weak/strong respectively instance/s of task each type in Ta</li></ul>                                     |   |             |           |                                    |   |  |            |            |           |                                      |  |   |
| Further Description                  | <table><tr><td>Constraint</td><td>Well formed</td><td>Malformed</td></tr><tr><td>Succeed<br/>((A), any(B))<br/>(weak)</td><td></td><td></td></tr></table> <p>The model on the right is malformed as an A is present but no B is present in the downstream of A (downstream here consists of tasks C and D).</p> <table><tr><td>Constraint</td><td>Wellformed</td><td>Malformed</td></tr><tr><td>Succeed<br/>((A), any(B))<br/>(strong)</td><td></td><td></td></tr></table> <p>The model on the right is malformed as two A tasks are present and the constraint requires a B in the downstream for each A, but only one is present.</p> | Constraint  | Well formed | Malformed | Succeed<br>((A), any(B))<br>(weak) |  |  | Constraint | Wellformed | Malformed | Succeed<br>((A), any(B))<br>(strong) |  |  |
| Constraint                           | Well formed   | Malformed   |             |           |                                    |   |  |            |            |           |                                      |  |   |
| Succeed<br>((A), any(B))<br>(weak)   |    |   |             |           |                                    |   |  |            |            |           |                                      |  |   |
| Constraint                           | Wellformed  | Malformed   |             |           |                                    |   |  |            |            |           |                                      |  |   |
| Succeed<br>((A), any(B))<br>(strong) |   |  |             |           |                                    |   |  |            |            |           |                                      |  |   |
| Examples                             | <p>Succeed weak ((plan exit, practical, interim, physio), all (assess))</p> <p>Succeed weak ((exit), all (plan exit, practical))</p> <p>The succeed constraint establishes a dependency between two sets. The presence of an instance of a triggering task type in P implies the presence of the target condition in the downstream of the triggering task. In this case:</p> <ul style="list-style-type: none"><li>If a plan exit, practical, interim or physio task appears in P, then assess task should appear before it in P.</li><li>If an exit task appears in P, then both a plan exit and practical task should appear before it in P.</li></ul>   |   |             |           |                                    |   |  |            |            |           |                                      |  |   |

Table 3-9 – The Resource Constraint

|   |   |   |            |           |   |  |   |
|---|---|---|------------|-----------|---|--|---|
| <i>Name</i>   | <b>Resource (Non-conditional composition constraint)</b>  |   |            |           |   |  |   |
| <i>Specification</i>  | <p>Parameters: (Ta)</p> <p>Attributes: Name</p> <p>In P, an actor can/must be allocated to at most/at least X task types in set Ta.</p> <p>In P, an actor can/must be allocated to at most/at least X task instances of anyone task type in set Ta.</p>   |   |            |           |   |  |   |
| <i>Further Description</i>                                  | <table><tr><td>Constraint</td><td>Wellformed</td><td>Malformed</td></tr><tr><td><b>Resource</b><br/>(C: (T - At most 1),<br/>(I - At most 1))</td><td></td><td></td></tr></table> <p>The model on the right is malformed as the human actor ‘Max’ has been allocated to two instances of a task of type C.</p> | Constraint  | Wellformed | Malformed | <b>Resource</b><br>(C: (T - At most 1),<br>(I - At most 1)) |  |  |
| Constraint  | Wellformed  | Malformed   |            |           |   |  |   |
| <b>Resource</b><br>(C: (T - At most 1),<br>(I - At most 1)) |   |  |            |           |   |  |   |
| <i>Example</i>  | <p>Resource (assess)- ((type value(at most 1), instance value (at most 1))</p> <p>By setting the type value to ‘at most 1’, it is asserted that an actor can be assigned to at most one of the task type targets. By setting the instance value to at most 1, it is asserted that any one actor can perform only 1 instance of the task type targets, in this case assess task instances i.e. assessments have to be performed independently by different professionals.</p>        |   |            |           |   |  |   |

### Language Relationships

The following table summarises permissible relationships between the concepts described in the previous section. All other relationships are prohibited. Precedence relationships establish ordering amongst tasks. Assignment relationships establish allocation (which actors are responsible for which tasks).

Table 3-10 – Constraints Links

| <b>Node</b>                               | <b>Relationships</b> |  |
|---|----------------------|--|
| <i>Task Type</i>                          | In                   | Target links in from include, exclude, precede, succeed nodes                                      |
|   | Out                  | Trigger links into serial, fork, include, exclude, precede, succeed and resource constraint nodes. |
| <i>Serial, Fork, Resource</i>             | In                   | Trigger links in from task type nodes  |
|   | Out                  | -  |
| <i>Include, Exclude, Precede, Succeed</i> | In                   | Trigger links in from task type nodes  |
|   | Out                  | Target links out to task type nodes  |

### 3.4.3 Procedural Models

Procedural models capture the design of the actual work which will be performed; the task instances, the relationships between task instances and the resources responsible for each task. P can be:

- *Unstructured* - A collection of tasks with no precedence relationships connecting those tasks. An unstructured procedural model can be likened to a checklist which

is used to express the tasks involved in the process and ensure they are done once and only once.

- *Structured* - A composition of tasks where precedence relationships amongst tasks have formed a directed graph. It is on such compositions that constraints established in D have most leverage.

### Language Concepts

*Start* - Each P can contain at most one 'start' which is a single beginning point to the process.

*End* - Each P can contain at most one 'end' which is a single end point to the process.

*Tasks* - Each P can contain multiple tasks. Tasks represent a manual (human) activity. Developing the approach, tasks could be used to represent automated tasks such as Web services (which often have the same requirements of flexibility and compliance in their orchestration as manual tasks have in process support). Tasks have several attributes:

- *Name* - The name of the task.
- *Type* - Tasks can be:
  - *Typed* - an instance of a task type defined in D (inheriting properties defined in D).
  - *Un-typed* - user defined, un-typed task for local needs (P can contain task that are not defined in D).
- *Join* - Each task instance contains join (conditions on when the task can be in the ready state) and split (conditions on what output edge are fired on completion/cancellation) behaviour. The user can select if the task should adhere to the 'AND' join or the 'OR' join:
  - *AND* - the task is set to ready (from waiting) when each of the its incoming edges is fired
  - *OR* - the task is set to ready (from waiting) when any one of the its incoming edges is fired

All task instances adhere to the 'AND' split only. Each edge leading out from the task is fired on completion of the task. Thus, P is more restricted than a typical procedural language.

- *Enactment State* - The status of the task in terms of work performance (see Section 3.4.4).

Business rules are often peppered amongst process design. This is seen in procedural modelling languages which offer different paths through the model during enactment through the ‘OR’ or ‘XOR’ split. The figure below shows a simple procedural model. The ‘assess’ task employs an ‘XOR’ split which means after it has been executed either the ‘cast’ or ‘sling’ task can be executed but not both.

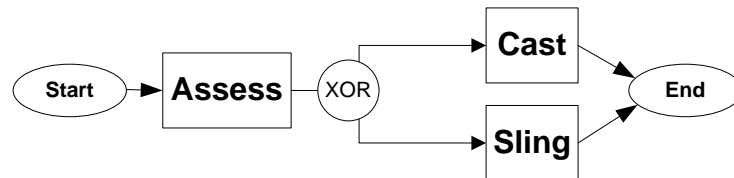


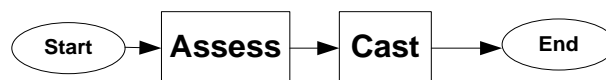
Figure 3-5 - Procedural Model with ‘XOR’ Split

This control-flow pattern increases the complexity of the model and makes it difficult to visualise planned work. In addition, advance verification of many of the constraints can not be established due to the uncertainty of enactment path.

A goal of this work has been to maintain a simple visualisable plan of what will occur and a simple record of what occurred. This is achieved by separating out two different types of content. The uncertainty of choice is represented in the declarative model rather than with the ‘XOR’ split. A resolution of that choice is made in the procedural model. The simplicity of procedural models and ability to verify a procedural model in advance is maintained as multiple pathways have been removed. Both P1 and P2 can be verified in advance of execution.

D1: Precede one ((A), (B,C))

P1:



P2:



Figure 3-6 - Declarative Statement with Procedural Resolutions

*Actors* - Each P can contain multiple tasks. Actor nodes represent the resources that are responsible for tasks. Currently the only attribute required is a name. Actor nodes are associated with task nodes through assignment relationships.

### Language Relationships

The following table summarises permissible relationships. All other relationships are prohibited. Precedence relationships establish ordering amongst tasks. Assignment relationships establish which actors are responsible for which tasks.

Table 3-11 – Procedural Links

| Node         | Relationships |  |
|--------------|---------------|--|
| <i>Start</i> | In            | -  |
|              | Out           | Precedence links out to task (unless the target task has a in-flow from a task node*)<br>Assignment links out to actor   |
| <i>End</i>   | In            | Precedence links in from task (unless the source task has a out-flow to a task node*)<br>Assignment links out to actor   |
|              | Out           | -  |
| <i>Task</i>  | In            | Precedence links in from task (unless the target task also has a in-flow from a start node*)   |
|              | Out           | Precedence links out to task (unless the target task also has a out-flow out to a end node*, or unless a path will be formed from the target task to the source task**)<br>Assignment links out to actor |
| <i>Actor</i> | In            | Assignment links in from tasks   |
|              | Out           | -  |

\* These conditions prevent unreachable tasks:

- A task is unreachable if it is the source of a precedence link whose target also serves as the target for a precedence link whose source is a start node.
- A task is unreachable if it is the target of a precedence link whose source also serves as the source for a precedence link whose target is an end node.

\*\* This condition prevents loops. Loops are problematic as ‘AND’ splits can result infinite loops. The procedural meta-model simplifies the handling of loops and repetition. Loops are represented in D. Their resolution is linearised by P; every repetition involves a unique set of tasks and flows.

### 3.4.4 Procedural Model Behaviour: Task Execution

Each task can be one of five enactment states (waiting, ready, ongoing, completed and cancelled). Figure 3-7 shows state transition which enables task execution.

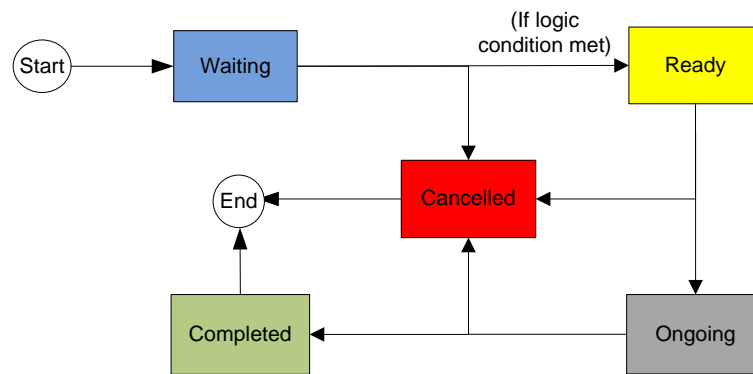


Figure 3-7 - Task Execution State Transition Diagram

- *Waiting* - The task is not ready to begin work on.
- *Ready* - The task is ready to begin working on at user's discretion. A task in this state may be more easily re-allocated to another person than one where the work has already started. Items that remain ready for a long time without being started, may point to a large backlog of the person responsible, and thus inform later allocation decisions.
- *Ongoing* - The task is currently being performed by the user.
- *Completed* - The task has completed normally.
- *Cancelled* - The task has been abnormally terminated/withdrawn from the plan.

These states are used to implement an enactment model for an individual task and P as a whole. Both the knowledge worker and automated enactment engine push the tasks through different enactment states. The 'waiting' to 'ready' transition is available when the tasks join condition is satisfied (precedence links which flow out from a task are fired when that task reaches the completed or cancelled state) or when the task has no preceding tasks (the task is the first task in a structured P or forms part of an unstructured P). All other transitions are unrestricted and driven by the knowledge worker. This aspect is described further in Section 4.5.4.

The enactment state of P as a whole can be derived by using the set of its task states. P is:

- *Ready* - For a structured P, if the first task is in state ready. For an unstructured P, if all tasks are in state ready (excluding tasks in a cancelled state).
- *Ongoing* - If at least one task is in an ongoing state.

- *Completed* - For a structured P, if the last task/s is in state completed. For an unstructured P, if all tasks are in state completed (excluding tasks in a cancelled state).
- *Cancelled* - If all tasks are in a cancelled state.

### Healthcare Scenario (Cont'd)

As described, P is a plan of work which can be checked for alignment with defined constraints. Figure 3-8 gives two (out of an infinite possible number) structured example resolutions of the D developed in the previous section. These example procedural models are examined for correctness in the next section.

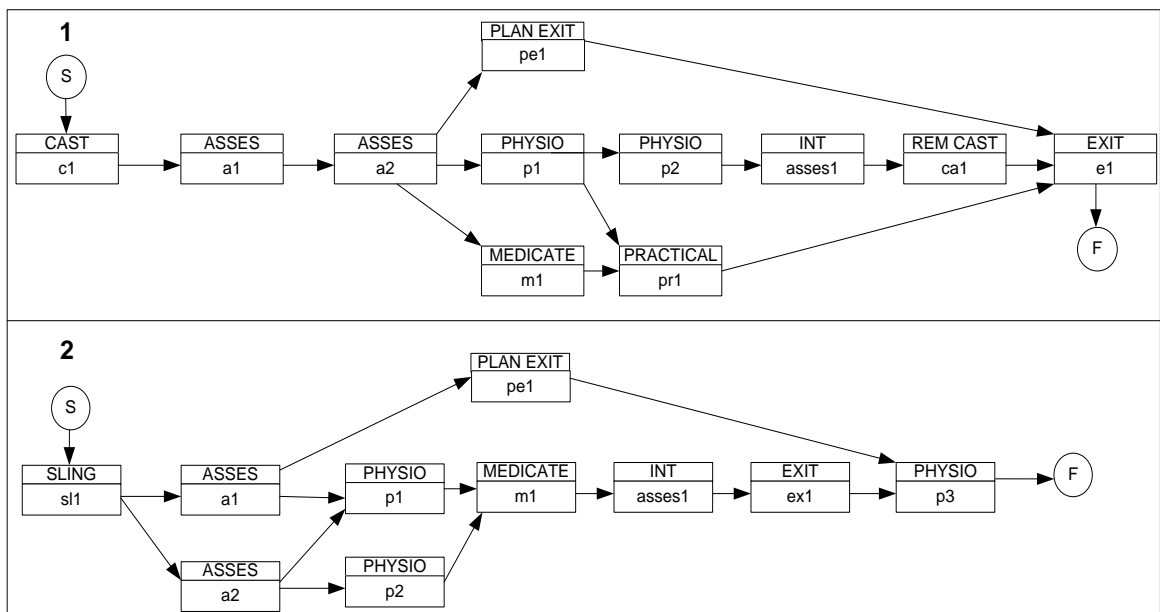


Figure 3-8 - Example Procedural Model

### 3.4.5 Computational Supports

On larger data structures, visual inspection can become difficult and time consuming. This is the rationale behind computational analysis tools which provide alternate views on data structure which emphasise certain aspects of it in a readily digested form e.g. tables and graphs. The view we focus on here is the verification view.



## Verification

The verification view examines a P to assign a value about the validity of P with respect to each constraint in D. P is in a sequence of states throughout its life. A snapshot is P at a moment in time. A verification view on P is always with respect to a specific snapshot of P.

The verification view then can be used to highlight problems which exist in the current composition of P. As P maybe modified frequently, the knowledge user can deploy the view to check that correctness has been maintained. The response to the identification of a problem (i.e. where P is not in alignment with D) is user driven. This view then contributes gaining and maintaining semantic correctness in work plans.

The approach is based on the assignment of one of five possible verification states to each constraint. The five verification states can be classed as well formed or malformed:

- *Well formed* -
  - *Well formed/Enacted* - P is well formed and task instances involved have been enacted (in completed or cancelled state).
  - *Well formed/Ongoing* - P is well formed and at least one task instance involved has not been enacted.
  - *Well formed/Not fired* - P is well formed through the constraint not being fired.
- *Malformed* -
  - *Malformed/Temporary* - P is not well formed. Domain requirements dictate if P can be corrected by user intervention i.e. the error is a temporary violation (due to something unknown and yet to emerge in the domain) or semantic error.
  - *Malformed/Permanent* - P is not well formed. P dictates that P cannot be corrected by user intervention as certain tasks have been enacted in error:
    - *Serial/Fork* - if more than one task instance of a target task type is enacted.
    - *Exclude* - if both a task instance of a triggering task type and a task instance of a target task type are enacted.
    - *Include* - if two task instances of different target types have been enacted and the logic mode is 'one'.
    - *Precede* - if two task instances of different target task types have been enacted and the logic mode is 'one'.
    - *Succeed* - if a task instance of a triggering task type is enacted and the target condition is not met.

- *Resource* - if more than the maximum (for type and type instance counts) has been enacted.

Pre-enactment, a malformed P can always be edited to bring it to a well formed state.

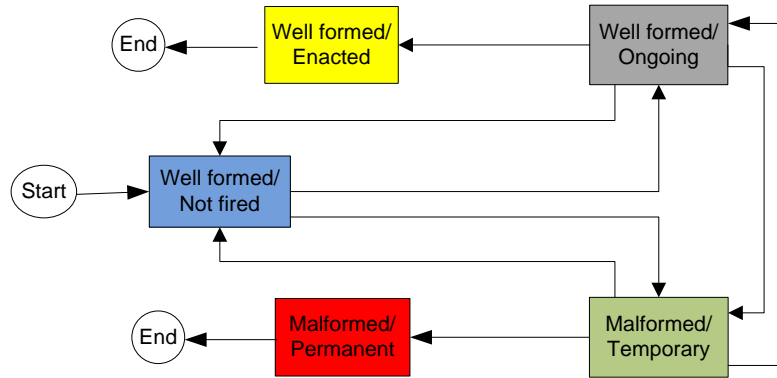


Figure 3-9 - Constraint State Transition Diagram

The state diagram above shows the possible states an individual constraint can be in at any one time. The figure above assumes an initially empty P. As the composition of P is modified, the state of a constraint can change. Simplified algorithms (generating well formed-not fired, well-formed and malformed) for each constraint are given in Appendix B.

Using the current states of each constraint in a D, a state can be assigned to D as a whole. Compositionality in this context means that it is enough to prove that a P satisfies each of the constraints in order to prove that P satisfies the model:

- *Well formed/Enacted* - When at least one of the constraints in D was fired by the composition of P, the constraint was not violated and P is in a completed state.
- *Well formed/Ongoing* - When at least one of the constraints in D was fired by the composition of P, the constraint was not violated and P is in a ready or ongoing state.
- *Well formed/Not fired* - When none of the constraints in D are triggered in P.
- *Malformed/Temporary* - When at least one constraint in D is temporarily violated by the composition of P.
- *Malformed/Permanent* - When at least one constraint in D is permanently violated by the composition of P.

Various options will be built into the way of working including handling of:

- *Mixed structure* - for each constraint, unconnected tasks will be treated as out path (i.e. not included in the up or downstream of any task instance).
- *Cancelled tasks* - for each constraint, cancelled tasks are discounted.
- *Empty trigger set* - the constraint will be treated as fired. Precede and succeed constraints are now equivalent to include.
- *Empty target set* - P will always be well formed with regards to the constraint.

### Healthcare Scenario Cont'd

Both of the procedural models developed in Section 3.4.4 can be evaluated in terms of each of the example constraints developed in Section 3.4.2. These models can be checked in advance of their execution.

Table 3-12 – Verification Summary for Example Procedural models

| Constraint   | Verification State      |                         |
|--|-------------------------|-------------------------|
|  | P1                      | P2                      |
| Serial (Pysio Session):  | Well formed             | Malformed               |
| Fork (Assess):   | Malformed               | Well formed             |
| Exclude ((sling), all(cast, remove cast)):                                 | Well formed - not fired | Well formed             |
| Include weak ((Medicate), all (Practical Session)):                        | Well formed             | Malformed               |
| Precede weak chain ((plan exit), all (exit)):                              | Well formed             | Well formed             |
| Precede strong ((exit), all (practical)):                                  | Malformed               | Malformed               |
| Precede strong ((Apply cast), any (physio session, prescribe medication)): | Well formed             | Well formed – not fired |
| Precede weak ((Assess), one(cast, sling)):                                 | Malformed               | Malformed               |
| Succeed weak ((plan exit, practical, interim, physio), all (assess)):      | Well formed             | Malformed               |
| Succeed weak ((exit), all (plan exit, practical))                          | Well formed             | Malformed               |

Each constraint which is malformed is temporary (as P has not yet begun enactment). Each constraint which is well formed is well formed-ongoing (unless specified as well formed-not fired). This information can be used by knowledge workers to target aspects of the composition of P which currently violate declared constraints.

### 3.4.6 Suitable Conditions for Use

It is worth noting that the approach presented above is suitable for EKP given the following conditions:

- This approach is leveraged most when there are a medium to large amount of possible compositions of P (i.e. loose to medium declarative models). If there were a small number of variants of P, the requirement of building P would be an unnecessary burden if there was no flexibility allowed in that build.
- The approach is geared for compliance-aware rather than safety-critical environments. This is due to the reliance on the knowledge worker to activate the concepts correctly.
- The approach is geared for low to medium frequency processes due the overhead of the composition of P.
- The approach is geared for low to medium complexity due to the lack of scalability of graphical representations; each type of content would become cumbersome to activate.

### 3.4.7 The Impact of Hierarchical Models on Approach

Using hierarchical process structures, work can be defined to an arbitrary depth. The grain of specification is domain dependant; users break down tasks to a depth which is useful. Hierarchical process structures allow users to expand on work description as it becomes known. For example, sub content may be defined for a 'Physio Session' task may be defined which includes a plan for the activities involved in the session such as stretching, hydro-therapy, massage, joint movement and muscle development tasks.

Hierarchical work definition allows clear distinction between categories of work; strategic planning are the top levels of work planning, tactical planning are the middle levels of work planning and operational planning are the bottom level. Different people can work on the relevant sections and levels of work description. The actual work involved in the composite task is the act of coordinating the sub-tasks it is comprised of.

#### **Impact on 'Pockets'**

A 'pocket' can be deployed as part of a hierarchical structure. The content of any composite task could be made up of P, P and D or D. The structure can be enacted. When a

constrained ‘pocket’ is reached it would need to be resolved (or already have been) into concrete procedural model.

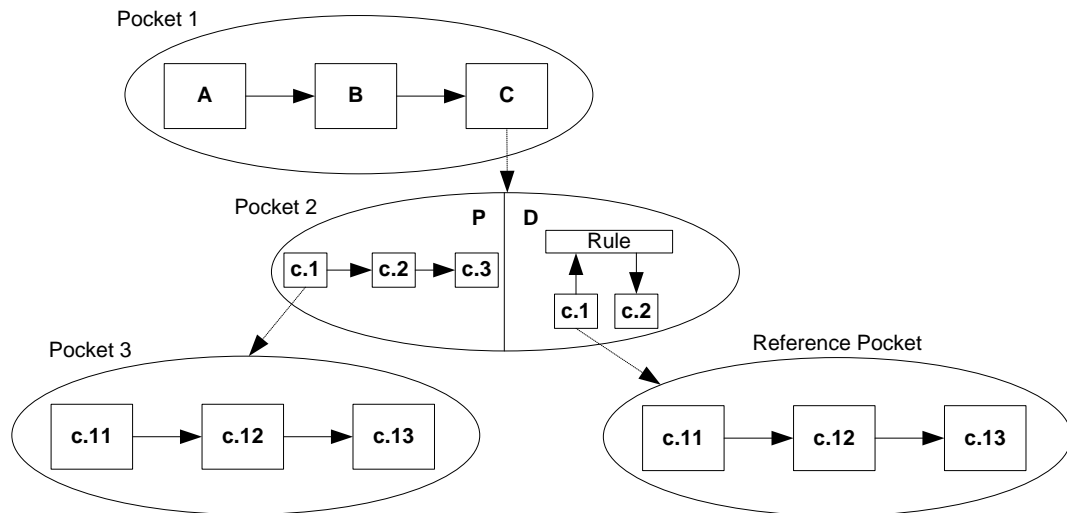


Figure 3-10 - Hybrid Models and Hierarchical Structure

The way of working would require hierarchical D models. Content of task types declared in D would form a ‘pocket’. On instantiation of a task type, its content would be copied over and available for use as part of P. Any P content declared as part of D is for reference only and cannot be enacted until instantiated. This way of working is visualised in Figure 3-9. ‘Task C’ of ‘pocket 1’ has sub-content defined. Declarative content has been resolved into procedural content. Task type ‘c.1’ of ‘pocket 2’ also has sub-content defined. This is the predefined procedural model of reference ‘pocket’. Each time ‘c.1’ is instantiated in ‘pocket 2’ its sub-content is also instantiated.

### Impact on Procedural Model Execution

These following operational rules are generally implemented in order to ensure that children are performed as parts of their parents.

- When a child of a task becomes *Ready*, the parent also becomes *Ready*.
- When a child is started, the parent is automatically started as well.
- When a parent is started, all children not waiting for input flows are made *Ready*.
- When a child finishes, and activates a flow to the output of the parent, if the output has received all necessary flows, the parent automatically finishes as well.
- When a parent is cancelled, all its children are cancelled as well.

## 3.5 Collaborative Working

User-model interaction can be facilitated by the PAIS in different ways (see Section 2.5.6). One form of user support which has received little attention is the facilitation of differing modes of group working in model interaction. In Chapter 2 it was established how PAIS can be strengthened by effective collaboration and wide participation. Group working is necessary to fully leverage the interactive modelling paradigm, making it an essential user support. This section will present a strategy for integrated group working.

### 3.5.1 Techniques to Support Collaboration

Successful participation and collaboration requires effective facilitation that can engage people, coordinate their joint actions and advance their common understanding. How can group-working modes be seamlessly incorporated into PAIS to assist in the performance process modelling and enactment?

#### **Modes of Working**

Group work will be facilitated firstly by providing different degrees of group work coupling. Coupling is the degree to which team member's act in the same time on the same artefact. The first level of group working is for all participants to access the same artefact at different times. The second level of group working is for all participants to access the same artefact at the same time:

- *Asynchronous* - A shared artefact is worked on at different times by multiple participants. This can be described as a 'staggered' access, with at most one participant working on the artefact at any time. For example, an orthopaedic doctor may initially create a rehabilitation programme for a patient which is amended by a physiotherapist at a later time.
- *Synchronous* - can be achieved in two ways:
  - *Tightly Coupled Collaboration* - A shared artefact is worked on at the same time by multiple participants. Participants work together in a shared workspace with a joint focus, shared navigation and control. This way of working is also known as real-time collaboration. For example, several orthopaedic doctors work together at the same time from their respective locations to create a rehabilitation programme for a patient.
  - *Loosely Coupled Collaboration* - Participants share the same set of artefacts and have the group awareness and activity awareness information available,

however, they have separated focus of attention and separated navigation and control to allow them work alone when they wish to do so at the time. For example, several junior doctors are examining various models created for previous patients independently but at the same time.

### **Planning Group Working Sessions**

There should be seamless shifting from individual work to group and vice versa. Two main ways of supporting the shift from individual to group working are:

- *Ad-hoc* - The process of initiating real-time collaboration can be spontaneous and informal. For example, a physiotherapist may quickly invite a bone specialist or junior doctor into his/her working session to get agreement or a second opinion about the design of a rehabilitation programme.
- *Planned* - The collaboration may be a more formal planned session/meeting with a pre-specified agenda. For example, a virtual meeting can be planned far in advance where senior doctors, surgeons and bone specialists meet to review current procedures.

### **3.5.2 Mechanisms to Support Group Working**

It is important that the higher level strategy of group working approaches described above is facilitated by lower-level technical protocols. Below, several key facilitators necessary to support group working are adapted from (Schummer and Lukosch 2007):

- *Meeting Room/Venue* - Provide participants a virtual working place with facilities for participants to meet and work together.
- *Login* - Make it as easy as possible for people to join collaborative activities by letting users identify themselves before they can use an application. This also allows intended participants to take part in the meeting and keep away those unintended.
- *Quick Goodbye* - Make it easy for a user to leave a group/session.
- *Group and Activity Awareness* - Provide information on who is present (participants of a session) and what they are doing.
- *The Stage* - The user interface which supports group working activities. It provides a joint focus for presentation or interactive collaborative activities allowing remote collaborators to work together over a visual work surface.

- *Direct Manipulation* - It is widely accepted that systems can provide effective support for cooperative work by offering a 'shared space' through which actors can interact directly. Participants can work simultaneously, where anyone can manipulate (create, move, resize, and delete) shared artefacts synchronously at any time.
- *Text Chat* - The key communication technique in groupware systems is text chat. Text chat can complement the direct manipulation of shared objects. Text chat should be closely incorporated into the shared space, so the chat and purpose of the chat are not fragmented.
- *Tele-pointer* - Refer to and focus participants' attention on a specific position/object on the shared space.

Chapter 4 provides further description on how these protocols are implemented. A comparison to other groupware applications can be found in Appendix C.

### 3.5.3 Collaborative Indirect Work

Group work can play an important role in gaining and maintaining the semantic correctness of models. Group inspection of model content can help identify errors in model composition. Group discussion and negotiation can help gain group consensus in how best to respond to the error and bring the model to a correct state. Each participant has their own perspectives, agenda and knowledge; partially understood, partially explicit, partially aligned and partially in conflict. The ability to articulate and understand each applicable viewpoint is essential for the support of EKP. Each viewpoint will be used to 'hammer out' a consensus of understanding and model content.

Developing models in real-time group work is particularly important in the early phases of distributed cooperation and model development, in order to quickly clear up misunderstandings and develop a shared understanding of work and agreement on model form. A shared understanding and agreement from participants helps avoid error and helps detect error in model composition both of which contribute to reducing re-work.

Let us further examine the role of group work in PAIS by developing the healthcare scenario. Table 3-12 summarizes the potential for group working for both procedural and declarative model content at all stages of the models life cycle.



Table 3-13 – Scope for Group Working

| Model Type         | Collaborative Activity  |
|--------------------|---|
| <i>Declarative</i> | <p>Create/modify:</p> <ul style="list-style-type: none"> <li>• Within a single organisation, multiple compliance experts may work together in a team. As a group, ideas could be generated and discussed to formulate how to best capture compliance rules in a declarative model. This could be a team made up of orthopaedic doctors, bone specialists and physiotherapists creating declarative content for the rehabilitation process.</li> <li>• For inter-organisational pursuits, representatives are required from each participating organisation to ensure that each respective compliance agendas are adhered to by the declarative model. This could be a team made up of a senior orthopaedic doctor and a government representative from the Department of Health.</li> </ul> <p>Analyse:</p> <ul style="list-style-type: none"> <li>• Informal analysis of the declarative model through direct group inspection and discussion to identify problems. It may be discovered that the current model does not adhere to recent changes in applicable legislation or codes of practise/ethics.</li> <li>• Formal analysis of the declarative model through computational supports used as a group to identify problems.</li> </ul> |
| <i>Procedural</i>  | <p>Create/modify:</p> <ul style="list-style-type: none"> <li>• Within a single organisation, multiple process experts may work together in a team. As a group, ideas could be generated and discussed to formulate the procedural model to best to meet the needs of the case at hand. This could be a team made up orthopaedic doctors, bone specialists, junior doctors and physiotherapists.</li> <li>• For inter-organisational pursuits, the team may consist of process experts from each participating organisation. As a group, the team can generate a procedural model which all the team agree on. This could be a team made up orthopaedic doctors, bone specialists physiotherapists and external health and safety officers.</li> </ul> <p>Analyse:</p> <ul style="list-style-type: none"> <li>• Informal analysis of the declarative model through direct group inspection and discussion to identify problems.</li> <li>• Formal analysis of the declarative model through computational supports used as a group to identify problems. Using the verification view, it may be discovered that the current model does not adhere to constraints.</li> </ul>   |

### 3.5.4 Further Anticipated Benefits of Real-Time Collaboration

The quality of process design in terms of efficiency (speed of development) may improve as time wasted by waiting for input from and agreement from other stakeholders can be minimised.

The quality of process design in terms of effectiveness may improve as participants are able generate, discuss and develop ideas in groups which would not have developed individually.

Furthermore, collaboration can not only be deployed for model building and adaptation, but also for related functions such as process learning/training. For example, an experienced user can guide a new team member through models. In terms of the healthcare scenario, group work could be leveraged by senior Doctors to train junior doctors by taking them through interesting past cases

## 3.6 Model Management

This section will extend the conceptualisation of interactive modelling approach presented in Section 3.3 by describing the impact of the hybrid modelling language and collaboration on how the interactive models can be managed on day-to-day basis. Typical examples of the life cycle of the various components which comprise a PAIS were given in Section 3.3. In this work we focus primarily on the management of the life cycle of the instance and type models as pictured in Figure 3-2.

A general overview of the model interactions required to push the type and instance models through their life cycle is covered. In doing so, the language described in Section 3.4 and opportunities for group-working described in Section 3.5 are drawn upon. The following sections will elaborate further with description of:

- Who will use the technology, how will they use it, when will they use it and in what capacity.
- Where rich opportunities exist for group working.
- Where the use of a hybrid modelling scheme has particular impact on model management.

### 3.6.1 Life Cycle of Type and Instance Models

At each phase of type and instance model management, process knowledge is continually encoded by any participants or stakeholders through manual interactions with graphical models. Each arrow in Figure 3-11 represents participant activity. Each activity will be covered in the description below organised into activities performed on the type, instance and completed instance models.

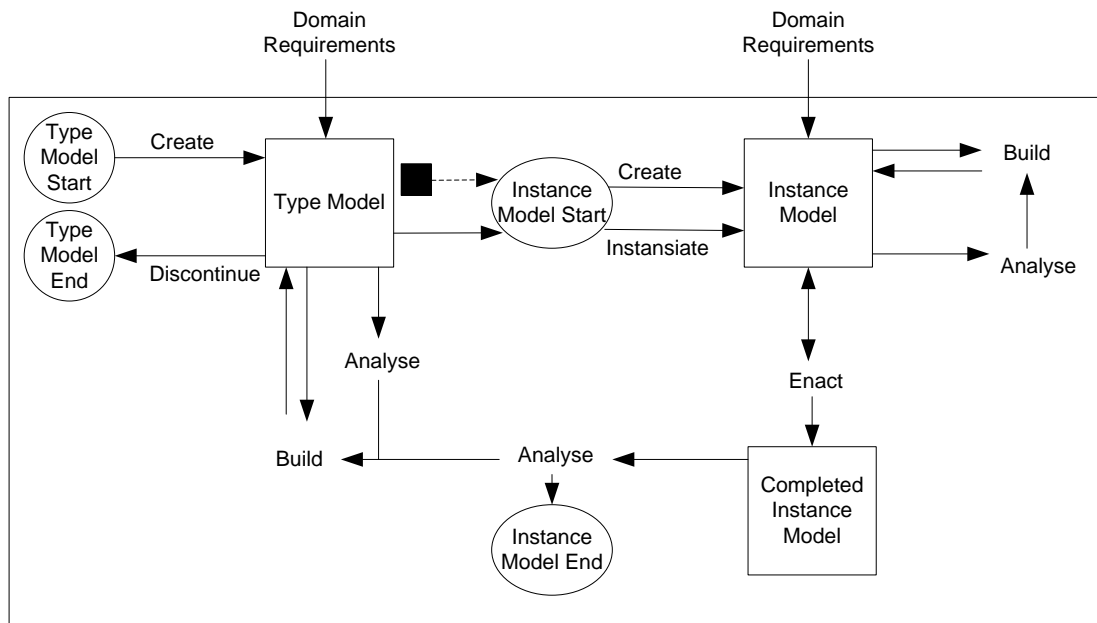


Figure 3-11 - Life Cycle of Type and Instance Models

### Type Model

The approach presented in this thesis is designed to support processes where a plan can be articulated and subsequently used to coordinate work. However, that plan often cannot be encoded in detail far in advance due to unknown requirements of the work (e.g. the design of later activities may be dependent on earlier outcomes or task allocations may not be decided until nearer the time of task performance) and large number of possibilities of plan composition (ruling out an approach where knowledge workers would make a selection from predefined designs).

The type model is used to accommodate these characteristics. The type model is used to drive future, usually recurring, work. A type model is created capturing domain requirements (best or preferred work practice); only the process knowledge which is known in advance which can be any mix of:

- Empty, partial and full procedural content.
- Empty, partial and full declarative content.

The hybrid modelling scheme described in Section 3.4 allows the user to make the right amount of specification at the right time (over-specify is avoided as asserting structure can be deferred until more is known closer to the time of enactment). This paradigm ensures recurring work is handled a flexible yet consistent way.

Before the type model is deployed, the quality of its design should be ensured as far as possible against domain requirements. If the structure is made up declarative content only, the model should be validated (its internal consistency ensured). If the model contains both procedural and declarative content the model should be validated and verified (consistency between P and D ensured). This analysis can be formal (e.g. verification view) or informal (e.g. activity can leverage group working through group model inspection.) Furthermore, past instance traces derived from the type model can be analysed to drive improvements in the type model. Both build and analysis activities can leverage group working.

A typical scenario would be for a type model to establish declarative content only which is to be resolved in to procedural content by each of its instantiations. This would be created by compliance specialists who are more aware of best practise and standards. Examples of type models which may form part of the healthcare scenario include patient admission, patient treatments (e.g. broken leg, broken arm etc), patient exit, recruitment and purchasing. A concrete example of a type model for a broken leg was developed in Section 3.4.2.

When the type model is no longer required in the domain, the model is discontinued. This means the model is removed from the type repository and is no longer available for instantiation.

## **Instance**

### *Create/Instantiate and Build*

The instance life cycle begins with either a type model being instantiated or the creation of an empty instance model.

- *Instantiate* - A type model can be instantiated as many times as required. For each separate work case, a user would create an instance of the type model (e.g. the broken leg type model) by cloning it. Initially, the instance model has predefined content which is a replica of and equivalent to its type model. At this stage, the instance model now is independent and changes can be made to both the declarative and procedural content as required by local conditions of work without affecting the type model (or other instantiations of it).
- *Create* - For non-recurring work a unique problem is addressed and therefore performed only once (a project for example). In this case the knowledge worker creates a new model with no initial content.

The process then executes on the basis of an initial, often partial model, where the full specification of the model is made at runtime (build activities provide the means of customising the process for that particular instance) and may be unique to each instance. The procedural content of a given instance model can be incrementally singled out from the space of possibilities defined by the declarative content as the process evolves. Section 3.4.4 developed examples of instance models for the rehabilitation type model. Each patient has an instance model which captures and plans the patient's treatment and rehabilitation programme.

The knowledge worker or groups of knowledge workers drive the process of dynamically composing the process through progressive definition/successive inscriptions made to the instance model to reflect and cater to domain requirements. Structure may be added or removed throughout the process, enabling users to plan their work with the level of detail that they find useful; any mix of empty, partial and full procedural and declarative models. Progressive definition is made through proactive user involvement and manual control rather than the reactive involvement of typical of workflow approaches.

The target of the build activity can be procedural or declarative content which is made through human (manual) interactions with the model. The type of change can be:

- *Fact-based* - What we know to be true (desirable or not) is reflected in model e.g. people frequently leave teams and resources may need to be removed from a task.
- *Design-based* - The change is not mandatory in a certain way but is designed e.g. people frequently join teams and new task allocations should be made.

It should be recognised that any modification is part of a wider system and ensured that any modification maintains the quality of each individual model and alignment between models.

#### *Enact*

Once procedural model design is fit for purpose it can be executed. See Section 3.4.4.

#### *Analyse*

The ability to identify and respond to problems as work progresses is highly sought and valued. The approach of progressive definition described above can be broken down to

cycles of problem identification and response to reflect emergent understanding of work requirements; some inadequacy in the model is identified through analysis and some part of the model is changed to address the inadequacy.

Through model analysis problems can be identified. Models can be analysed from two perspectives; analysis as governing content (assess alignment with what the structure governs) and analysis as governed content (assess alignment with governing structure). Table 3-14 and 3-15 show the impact of the two types of content from the perspectives of governed and governing structure.

Table 3-14 – Models as Governed

| <b>Governed – assess against superior structure</b> |   |
|---|---|
| <i>Procedural</i>                                   | Check model is in alignment with governing forces: <ul style="list-style-type: none"> <li>• Domain requirements - what the model should achieve.</li> <li>• Related declarative model - adding and rearranging tasks and task precedence, actor and actor allocation can cause an invalid model by violating a constraint. This can be assessed through verification. If an error is found, domain requirements dictate response.</li> <li>• Meta-model - for process modelling.</li> </ul> |
| <i>Declarative</i>                                  | Check model is in alignment with governing forces: <ul style="list-style-type: none"> <li>• Meta-model - for declarative modelling.</li> <li>• Domain requirements - what the model should capture.</li> </ul>  |

Table 3-15 – Models as Governing

| <b>Governing – assess against inferior structure</b> |   |
|--|---|
| <i>Procedural</i>                                    | Check impact on lower levels of work decomposition. If the intention is changed, its realising structure may need to be adjusted to achieve it.   |
| <i>Declarative</i>                                   | Check impact on any related procedural model. It particularly should be checked if the declarative model can be satisfied. It is possible that satisfaction can never be achieved due to the current state of the procedural model.<br>In some cases, it may be necessary to change declarative content to meet domain requirements although a related procedural content is already executing. This could be to relax or strengthen the model (i.e., add or remove activities, constraints, their attribute values and their associations) |

This way of working allows process-oriented quality management in change as the hybrid modelling scheme and group working contribute to error prevention. In addition, this way of working allows product-oriented quality management in model change as defects can be detected and corrected through user supports such as group working and verification views.

### Completed Instance Model

When the instance model is in a completed state it is removed from ongoing work list and stored in an instance repository. A desirable by-product of a process-centric approach is a

permanent record of the work which actually occurred. Such records are useful for future analytical and accountability purposes. For example, instance model content can be analysed to help improve its type model design. In terms of the healthcare scenario, a patient rehabilitation plan may have been the subject of a complaint, on analysis the procedural content adhered to relevant constraints, yet the complaint was upheld. This could be used to update the declarative content to prevent the situation from arising in the future. Alternatively, if the work was not type model-driven, a type model may be extracted from the instance for future use.

### 3.7 Summary

This chapter has introduced a framework for a novel approach to the flexible support of EKP. The framework is made up of a conceptualisation and implementation strategy. The first two critical areas of EKP (achieving a balance between process flexibility and process control along with addressing the potential for group working in process aware environments) have been addressed by the conceptualisation presented here.

The interactive modelling approach (Jørgensen 2004) to process support, introduced in Chapter 2 was further conceptualised and extended. Interactive models use visual models of work, which are created and consumed by teams of knowledge workers, to drive their day-to-day work. The interactive modelling approach has been extended by underpinning it with a hybrid modelling language. It has previously been implemented using a procedural language only. The hybrid language described in Section 3.4 combines a declarative constraint language and a traditional procedural language. Constraints are used to restrict the composition of a traditional procedural model. Wrapped around the hybrid interactive modelling approach is the ability to work in groups. This is becoming more important due to the need to leverage skilled human resources wherever they may be located. The final section gave a methodology for how interactive models can be managed given the impact of a hybrid modelling language and the potential for differing modes of group working.

The two remaining critical areas will be tackled by choices made in the implementation of the conceptual design, which forms the content of the next chapter.

# Chapter 4

---

## Implementation: Realisation of Approach in Web-based Environment

Chapter 3 presented a conceptualisation component of a framework which addresses the flexible process support of EKP. This conceptualisation consisted of a deployment method (extended interactive modelling), a language specification (a hybrid modelling language) and a user support (a strategy for group working opportunities). The exploitation of the strengths associated with the conceptualisation presented in the previous chapter is tightly bound with the availability of a software support which can realise its concepts. This is the final component of the integrated framework introduced in Chapter 3.

The Web-based collaborative knowledge management system PowerMeeting is selected to serve as a test bed for the hybrid interactive modelling approach. A description of the PowerMeeting system and the rationale behind its use as a test bed is given. PowerMeeting is extended with the D2P groupware application. D2P supports collaborative creation, verification and adaptation of graphical procedural and declarative models. The process of extending PowerMeeting with the D2P groupware application is summarised. A walkthrough of both systems is also provided.

The combination of PowerMeeting and D2P fully realises the conceptualisation presented in Chapter 3. A working academic software prototype which serves as a proof of concept then is the first form of evaluation. The healthcare scenario used in Chapter 3 will be developed in this chapter to demonstrate the capabilities of D2P. The chapter closes with a summary of the approach.

### 4.1 The Nature of Academic Prototypes

Academic software prototypes are established to explore specific research objectives. The prototype is geared towards the testing of novel and significant claims. The prototype can serve purely as proof of concept and it can be used as a platform on which to base further



evaluative studies. The purpose of the prototype is then highly targeted; it does not represent the development of a full industrial strength system.

A prototype has been developed which can test the suitability of a hybrid modelling approach to support EKP. Chapter 6 indicates areas which would have to be developed to turn the prototype into an industrial strength tool.

## 4.2 High Level Requirements

The rationale for the form of the conceptualisation presented in Chapter 3 can be found in the organisational requirements which emerged through analysis of the problem domain summarised in Chapter 2. The majority of the rationale for the form of the implementation can be found in the conceptualisation. Some deficits identified in Chapter 2 are however addressed directly by choice of implementation, in other words, the choice of implementation itself contributes to the needs of flexible support for EKP.

For the purposes of this discussion, it is useful to identify three distinct types of user who interact with a PAIS:

1. *Knowledge Workers*: End users of the system who develop domain models (type and instance models) as part of their day to day work.
2. *Application Developers*: Creators and adapters of applications. These are professional users who develop meta-models, computational supports that analyse model content and user interfaces to consume and visualise those services.
3. *System Developers*: Responsible for development of the underlying system services such as persistency and collaboration management.

The needs of user types 1 and 2 have primarily driven choice of implementation strategy. Requirements have been derived from the deficits identified in Chapter 2.

### **Knowledge Workers**

It was found that current PAIS lack accessibility. The system should:

1. *Be Easily Installed*: It should be quick and simple to start using the tool. The client installation process should take minimal effort (or no installation is required at all) to aid rapid connection to work environment.

2. *Be Readily Available:* Access to the tool should not be restricted to specific locations and machines. The system should facilitate the leveraging of users from across the organisation (geographically and hierarchically) wherever they are and with whatever machine they have access to.
3. *Have Satisfactory Performance:* Performance should meet or exceed user experience of comparable (graphical modelling and group working) technologies.
4. *Be Widely Compatible:* The system should be compatible with various and emerging platforms, different client access devices, different client access software and network bandwidths.

It was found that current PAIS are often fragmented tool sets. The system should be a single, integrated system to support a range of and shifting specificity.

### **Application Developers**

In addition to requirements derived directly from identified deficits, the following requirements are important in pursuit of flexible PAIS:

1. *Rapid Development:* Developers must be able to quickly and simply build or modify an application to respond to support needs as they emerge.
2. *Sustainable Environment:* The underlying platform should be scalable and readily tailorable. Ability to vary capabilities and behaviour as experience builds and requirements emerge is vital.
3. *Rapid Deployment:* The tool should be quick and simple to roll out.

## **4.3 Guiding Principals**

From the above requirements we can begin to draw out guiding principles which will help address those requirements.

### **Knowledge Workers: The system should be Web-based**

It has already been shown how companies are looking for more effective ways of doing business. One way which will set companies apart is how effectively they leverage Web technology to enhance their business and drive greater results. Web technology can be employed to meet the requirements of knowledge users.

There is a current tendency towards Web-based applications and tools. The Service Web is a term coined to describe professional business applications which are Web-based. Web technology can be harnessed to play a far stronger role in the design and delivery of PAIS than what is does at present.

The main advantage of Web-based tools is that they can be accessed from different computers just by using a Web browser, without the need for installing any further application (requirement 1). Moreover, ubiquitous access to these applications also means that the data resulting from their use does not need to be stored locally, but in servers globally accessible (requirement 2). Compatibility is also facilitated as Web-based applications can be configured to work with different software (e.g. browsers) and hardware (e.g. devices) (requirement 4).

Professional, business related Web applications have not taken off until recently. The barriers (poor performance in publishing to the Web, wide participation and rich user experience) have been removed through better technical facilitation. The functionality and performance associated with the desk-top is beginning to be replicated over the Web (requirement 3).

A final benefit of using a Web-based implementation is the scope for better take-up of groupware tools and PAIS tools. Ease of engagement to such tools can be achieved by making the collaborative applications widely accessible and easy to use. Using standard Web browsers as a front end can be used to offset the relatively poor take-up of groupware technologies as Web browsers are widely available and familiar to a wide and increasing number of people.

**Application Developers: The system should leverage a readily extensible, generic modelling environment**

The requirements of agile organisations (flexible software systems which can support shifting and emerging forms of organisation and processes) are in increasing conflict with traditional forms of software development.

A key theme has emerged in software development which addresses the need for rapid development and solutions; customisation of generic environments through component integration (use of software components provided by third parties).

Generic environments provide a basic framework usually with basic, reusable shared services. These services are not directly consumable, but used by applications. Specialised components can be quickly developed and plugged into the framework to leverage its services (requirements 1, 2 and 3).

## 4.4 Web-based Synchronous/Asynchronous Groupware

It was described in Section 3.5 how the approach to flexible process support leverages both synchronous and asynchronous group working. The most challenging aspect of realising the conceptual design, given the principles above, is the requirement for synchronous group work with a rich, intuitive user experience over the Web.

As shown in the related work section, numerous examples of task specific, real-time synchronous group working applications with rich user experience can be found including conferencing, shared editors, gaming, instant messaging, workflow and project management systems. However, synchronous groupware tools have yet to gain a strong Web presence (tools using standard Web browser front-ends are still rare (the above applications sometimes leverage the Web asynchronously)). XCHIPS (Wang, Finch et al. 2009) implemented a interactive modelling approach which focused on group working however it was delivered as a desktop application.

Lack of web presence could account for the poor take-up of such tools (Baker, Greenberg, et al. 2002). Indeed, gaming and instant messaging are the synchronous tools which have enjoyed considerable take-up arguably due to their Web presence (however, these examples are not fully fledged synchronous groupware applications with task specific support and rich user experience).

The technological issues which have accounted for the lack of Web-based synchronous groupware are described in Appendix E. Rapidly developed Web-based synchronous groupware systems have become a reality due to recent technological and methodological advancements. PowerMeeting is an existing framework which fulfils the implementation strategy given above. The PowerMeeting framework is described in Section 4.4.1. It is necessary to understand certain aspects of PowerMeeting to give context to D2P. The interactive modelling approach, hybrid modelling language and model management

strategies are realised by the D2P groupware application, which is delivered as a plug-in to the PowerMeeting system, is described in Section 4.5.

#### 4.4.1 The PowerMeeting Framework

The PowerMeeting framework facilitates the development and delivery of Web-based real-time, shared-workspace groupware (Wang 2008). PowerMeeting is a hub providing the necessary underlying collaboration and data management services for collaborative applications on the Web.

PowerMeeting can be described as a hub as it serves as a central and common port where various task specific groupware applications can be 'plugged in' to leverage its Web presence and group-working facilities. Each application is designed for some business, social or academic purpose. Being a Web-based service, PowerMeeting supports multi-channel access (software and hardware) to applications and application data.

Rich Internet Applications (RIA) are centred on user contributed Web content and a richer user experience through advanced interface widgets more traditionally associated with desktop applications. PowerMeeting and its plug-ins conform to this paradigm where users can collaboratively create, adapt, view and analyse graphical content with desktop-like responsiveness.

Perhaps the main limitation of a Web-based tool is the overhead imposed by translation and network (i.e. the response and performance times). By combining this constraining factor with an utmost concern for user experience, a governing notion can be formed. In this environment, the focus is on development of lightweight applications targeted to user task and less tolerant of feature creep as in desktop applications. So despite being complex applications with a strong backend, each application is considered as lightweight regarding the interactions of users.

Figure 4-1 presents a novel high level view of the overall architecture of the PowerMeeting system.

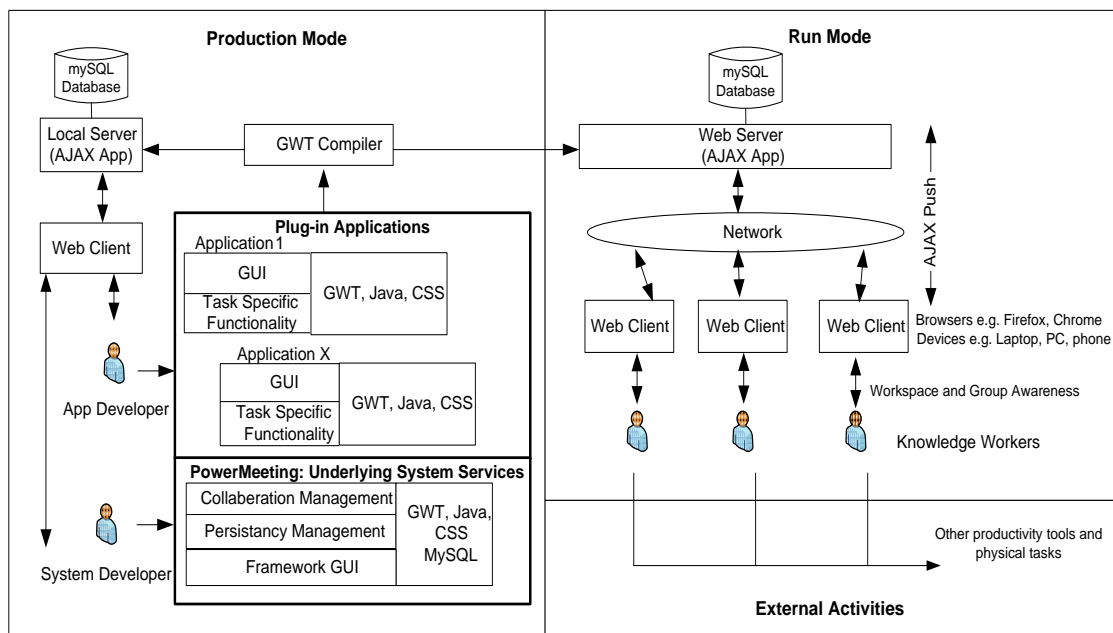


Figure 4-1 - Overarching Architecture for PowerMeeting

Figure 4-1 shows both production and run modes of the PowerMeeting framework. In production mode, the developer can create and modify applications and test them rapidly by running PowerMeeting on a local server. Once the system and applications are in a suitable state, a live instance of PowerMeeting can be run on a Web server for wider consumption.

Specific components of the framework are now described.

- PowerMeeting: Underlying System Services** - The main services provided at the system level are collaboration (user, session and replication) and persistency management. To ensure consistency across clients involved in the session, users and data involved in a synchronous working session require dynamically replicated shared data as well as transactional support for access to and modification of this shared data. PowerMeeting uses an AJAX-driven approach to overcome the problems described further in Appendix E. Persistency refers to database management to ensure data generated in the session beyond that session (applications and application data are not lost). The hub is visualised in a GUI, described in the next section.
- Plug-in Applications** - The requirement for a sustainable development indicates a component-driven environment where components can be rapidly developed and readily integrated to vary capabilities and behaviour of a system as experience

builds and requirements emerge. Various groupware applications can be developed and inserted into the PowerMeeting framework. Each application is geared for some task specific need. For simple development of Web-based Groupware Applications PowerMeeting combines GWT with the Commonground framework. Developers need only consider the underlying logic and user interface of the application. Refer to Appendix E for further detail on the use of GWT and Commonground.

- *Network* - The low level physical and software systems which facilitate data transfer.
- *Web Clients* - Standard Web browsers displaying HTML files are used to consume the services provided by PowerMeeting.
- *Other Productivity Tools* - As part of their work, users may use productivity tools such as word processing or spreadsheet applications.

### **Summary**

PowerMeeting is a suitable test bed as it supports the synchronous collaborative working with a rich user experience through direct manipulation of graphical artefacts required by the conceptual design of Chapter 3. Furthermore it supports the implementation requirements of user type 1 as it is Web-based. The only common skill requirement is the ability to use a Web browser. The approach works in all major browsers without need for installation ensuring a wide potential for adoption.

The requirements of user type 2 are also supported. PowerMeeting facilitates rapid development of synchronous Web applications as the only skill required is basic knowledge of the Java programming language. PowerMeeting meets sustainability requirements through use of plug-in applications.

### **4.4.2 PowerMeeting: System Walkthrough**

The front end of PowerMeeting is visualised as a Web page in a Web browser with a graphical user interface establishing the work environment. On browsing to the PowerMeeting Web site, the front-page is displayed. A login is required consisting of a user name (users will be known by this name for the duration of the session in the room they log into) and the name of the room they wish to use (this can be a new room or existing).

Rooms are virtual work areas where users can work in groups or independently. An infinite amount of rooms can be created. Each room is identified uniquely by its URL. A room could be created for each department of a company or time period to organise work for example. Documents are previously created work artefacts. Each room may contain multiple documents. A single document would represent the treatment plan for a particular patient for example.

As can be seen in Figure 4-2, the user interface for each room is the same and is divided into system level (housing services available to each application such as document management, document lists, chat and session participant lists) and application level areas (a container holding the user interface for the groupware application and data of the currently opened document.)

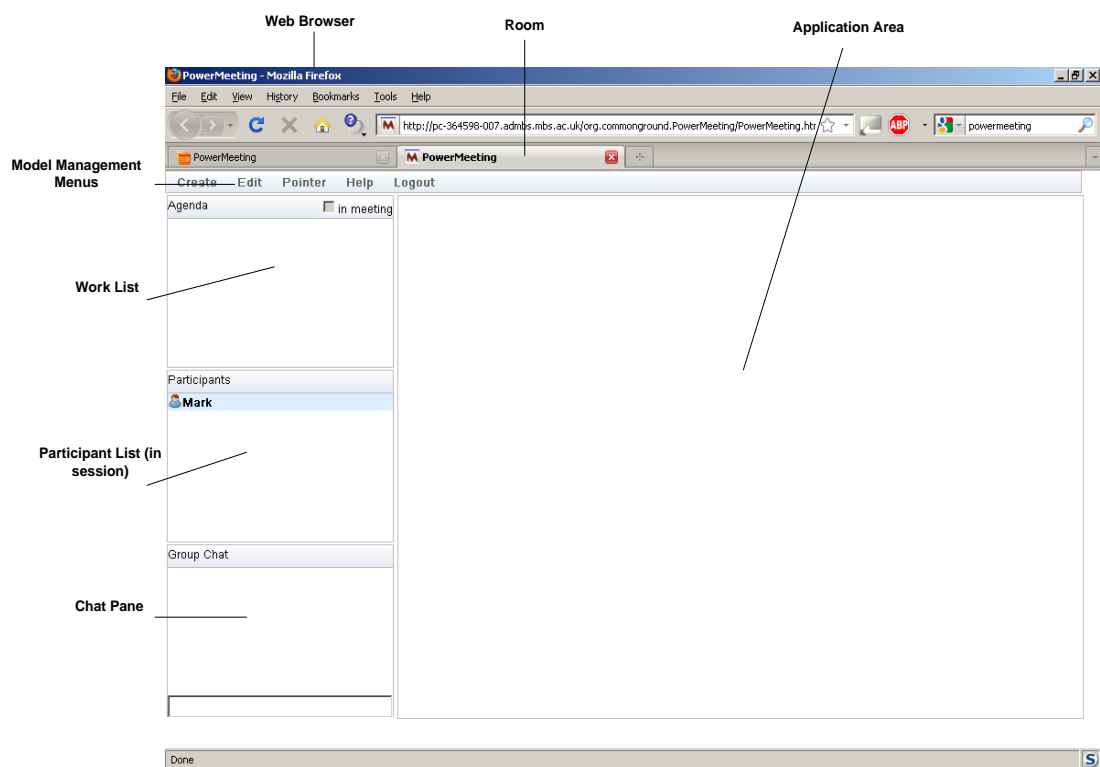


Figure 4-2 - PowerMeeting GUI

## System Level

As can be seen on the left hand side of Figure 4-2, the bottom panel contains a chat function, the middle panel contains a list of users who are active in the session and the top



panel contains a work list (documents associated with the room). On selecting the document, the relevant groupware application will open (along with the data for that document). The model management menu above the top panel allows common tasks such as create, edit and remove documents from the work list in addition to logging out.

Each groupware application has common tools to facilitate group working, important for fine-grained coordination. The two main tools available to each application are:

- *Chat* - Session users can communicate in natural language. We use a linear style for text chat. Participant's messages are added in temporal order to the end of a single transcript. No special fields are included except for the name of the contributor.
- *Pointing* - For use in tightly coupled collaborative sessions, a shared marker which all participants can see to focus attention on specific objects in the model.

### **Application Level**

The right hand area forms the main shared work space (where applications and application content is visualised and manipulated). The task specific user interface for each application is fully contained in this area. Users can exit a room by logging out or closing the browser or navigating the browser to a different page.

### **PowerMeeting Terms**

Table 4-1 gives a summary of the terminology used by PowerMeeting.

Table 4-1 – PowerMeeting Terms and Definitions

| <b>Term</b>              | <b>Definition</b>  |
|--------------------------|--|
| <i>Front Page</i>        | A Webpage providing a login facility to the PowerMeeting system. This Webpage serves as the front door. <ul style="list-style-type: none"> <li>The various medical professionals would browse to this location and login to begin using PowerMeeting</li> </ul>  |
| <i>User name</i>         | Each participant logs into PowerMeeting with a username. The participant will be known by this name for the in the room for the duration of the session. <ul style="list-style-type: none"> <li>The various medical professionals would be assigned a username. The username is required to login to PowerMeeting</li> </ul>                               |
| <i>Room</i>              | A virtual workplace. Multiple rooms can be created, which are then available to be accessed. Multiple documents can be associated with each room.  |
| <i>Session</i>           | The time spent in a room by an individual or groups of individuals.  |
| <i>Application</i>       | A groupware tool which has been plugged into the PowerMeeting system and is available for use e.g. calendar, drawing and voting applications.  |
| <i>Application List</i>  | The set of applications which are available to use in PowerMeeting system.   |
| <i>Work list</i>         | List of documents associated with the room. The list can have documents which open with different applications. <ul style="list-style-type: none"> <li>A list of pointers to multiple documents which could represent the current patients in the rehabilitation department.</li> </ul>  |
| <i>Document</i>          | Work created and available on the work list. Each document is created with one groupware application. That document will open with that groupware application. A document is therefore is equivalent to an instance model. <ul style="list-style-type: none"> <li>A single document could represent the treatment plan for a particular patient</li> </ul> |
| <i>Document Template</i> | A document with predefined content which can be instantiated (copied) and added to the work list. A document template is therefore is equivalent to a type model. <ul style="list-style-type: none"> <li>A single document could represent the generic treatment plan which can be copied and customised for the needs of a particular patient.</li> </ul> |

## 4.5 The D2P Groupware Application

The concepts described in Chapter 3 have been operationalised by a groupware application called D2P which can ‘plug in’ to the PowerMeeting environment. In realising our approach with this system we contribute to Web development by extending the kinds of application to which Web technology has been applied. It is worth emphasising that PowerMeeting provides a generic collaboration environment. D2P extends and uses that environment to implement a novel approach to flexible process support.

D2P implements an interactive modelling approach at the type and instance levels to flexible process support. The figure below shows the basic architecture of the D2P tool through its currently implemented model interactors which work on D2P documents. Thus when used with PowerMeeting, D2P builds on XCHIPS firstly by delivering a interactive modelling approach as a web application and secondly by delivering a hybrid modelling language.

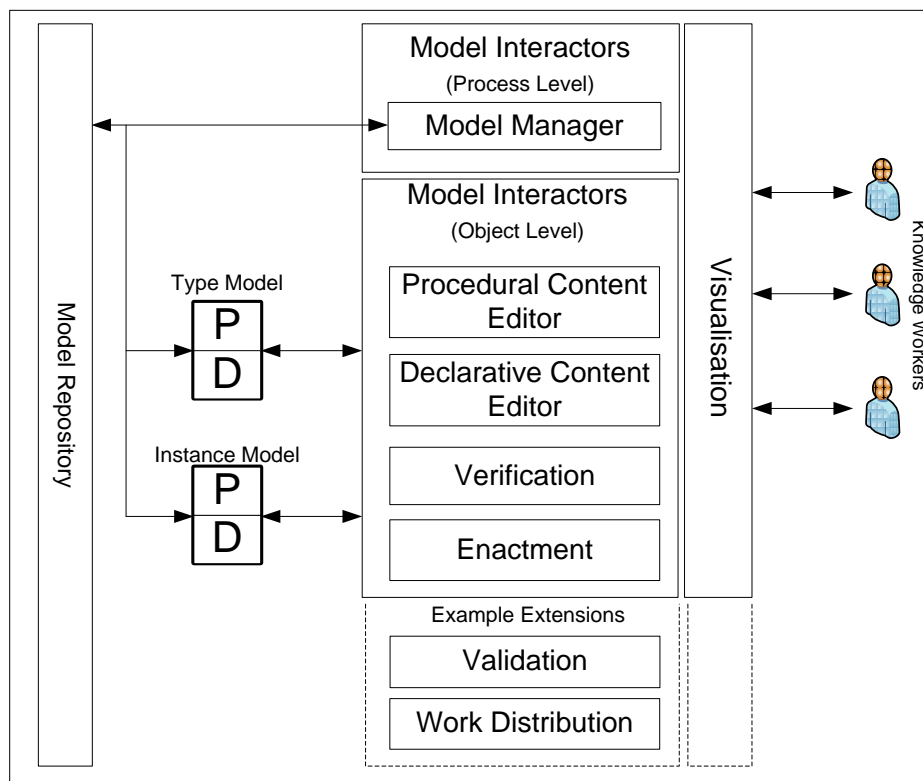


Figure 4-3 - D2P Architecture

At the centre of interactive modelling approach are the interactive procedural and declarative models which capture process logic. Through several model interactors, (e.g. editors, enactment and verification) knowledge workers can create, adapt, analyse and execute type and instance models. Model interactors form part of the application logic. The next three sections describe each of these model interactors. Each of the model interactors is made up underlying logic and an interface component for visualisation. The set of model interactors is extensible, for example the diagram shows a validation view could be developed in the future.

The help documents for the D2P groupware application can be found in Appendix H.

### 4.5.1 Model Manager

D2P documents can be created in one of two ways:

1. *New Document*: A blank document is created. A type or instance model can now be created by opening the document with the D2P model editor.
2. *Type-based Document*: A document with predefined content is created. An instance model has now been created. Work can begin by opening it with the D2P model editor.

These are accomplished using system level menu operations as seen in Figure 4-2. Once the document is created, a link to it is placed in the work list. Selecting the document from the work list opens the document in the application area.

### **Model Manager Primitives**

D2P enables the following atomic user actions:

- *Create new document (on 'create' menu select 'New', then 'create D2P').*
- *Instantiate existing type document (on 'create' menu select 'Predefined', then select the name of the predefined document).*
- *Delete document (in edit menu select delete (deletes currently selected document)).*
- *Rename document (in edit menu select rename (rename currently selected document)).*

### **4.5.2 Procedural and Declarative Model Editor**

D2P splits the workspace into a procedural modelling editor and a declarative modelling editor. Each editor is made up of a pallet and a modelling space (where models are visualised and manipulated). Each modelling pallet is made up of a node type menu (available node types e.g. task, actor etc to be instantiated in the modelling space) and view menu (available views e.g. verification view, help documents). All menu items are visualised with a symbols and text for the node/option.

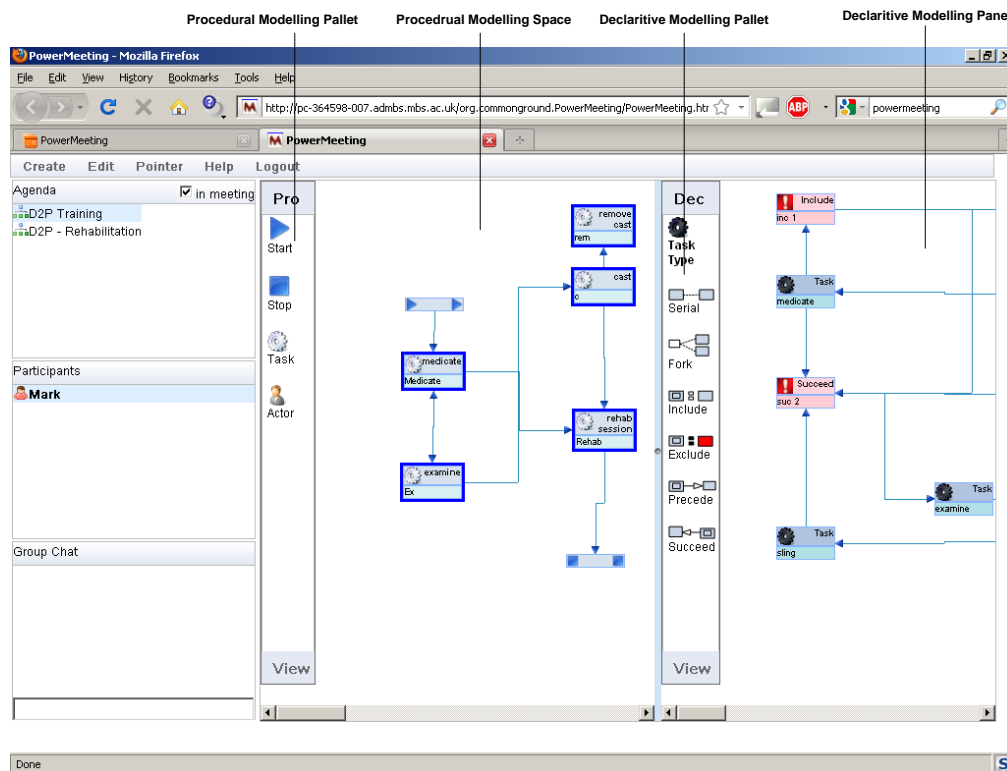


Figure 4-4 - Screenshot of D2P and the PowerMeeting Tools

Both type and instance models are visualised and manipulated in cooperative model editors/browsers. The required frequent modifications of the graphical models are made manually through direct manipulation with a rich user interface. Aspects of the rich user interface include drag and drop modelling which are not often found on the Web.

The size of each modelling area can be adjusted depending on the focus and current task of the user e.g. if the declarative model was being inspected it could be resized to fill the majority of the application space. This flexible space helps with user task focus, for example, in recurring work experienced users may soon get to know the constraints and can choose to minimise/hide the constraint model.

The current implementation is limited to a flat structure on one page (i.e. non hierarchical). Implementing hierarchical modelling would require a navigational mechanism to view node content. This could be handled by the groupware application itself or by extending the functionality of the work list to handle nested documents (a tree).

## Language Implementation

Chapter 3 described an implementation independent language as part of a conceptual approach for flexible process support. The language defined a knowledge representation scheme for procedural and declarative models. The scheme is essentially made up of concepts, relationships and semantic constraints establishing which concepts can be connected to which concepts through which relationships. The challenge of implementation is how to realise the language described in Chapter 3 in a working modelling system.

A graphical interpretation of the knowledge representation scheme has been selected to help ensure the hybrid modelling techniques recently employed in the workflow domain can meet the human requirements of the interactive modelling approach (ready articulation and interpretation).

Each type of content (declarative and procedural described in Sections 3.4.2 and 3.4.3) is a semantic net made up of concepts which are visualised as nodes (labelled icon images) and relationships which are visualised as links (arrows) between nodes. Languages based on boxes (nodes) and arrows (relationships) are supremely dominant in IS for mid to high level knowledge representation (low level knowledge representation usually employs textual programming languages such as Java, VB, XML etc). Such graphical and visual knowledge representation has rapidly gained acceptance for showing the interconnectedness of elements. Graphical representation affords flexible modelling as information can be established in different ways:

- *Navigational Links* - Established by explicit edges between nodes.
- *Spatial Links* - Established by spatial arrangement of nodes e.g. proximity.


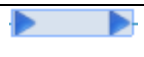





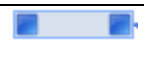
## Node Design

Each of the different concept types described in Sections 3.4.2 and 3.4.3 are implemented as node types. Each node type can be instantiated as required. The node instance is then visualised in the modelling space. For example, the generic concept of an actor forms a node type which can be instantiated to form a concrete representation of a real world actor in the modelling space. Various node instances of differing node types (and links connecting those node instances) build graphical procedural and declarative content for a D2P document.

Our approach to process modelling is to incorporate concept-related attributes, such as state for tasks, into node type definition. Each node model supports operations to perform object level (e.g. delete node) and attribute level (get and set attributes) operations. Each node type has a menu tailored for its needs (e.g. only the task node has the menu option to set state).

The pallet symbol and node design of each the four procedural nodes (Task, Actor, Start and Stop) are shown in Table 4-2. The user would click the pallet symbol to create a node on the graphical modelling surface.

Table 4-2 –Procedural Node Summary

| Node Type    | Pallet Symbol   | Node  |
|--------------|---|---|
| <i>Start</i> |    |    |
| <i>Task</i>  |    |    |
| <i>Actor</i> |   |   |
| <i>Stop</i>  |  |  |

The node type icon in the upper left panel of each node serves as the drag handler. Right clicking on any part of the node outside of the drag handler will show the menu for the node.

For the task node, the upper right panel displays the name of the task type (left blank if un-typed) and the bottom panel displays the name of the task node instance. For example the task type may be 'Examine' and the task node instance may be 'examine 12/03'. For the actor node, the upper right shows the name of the name of node type (Actor) and the bottom panel shows the name of Actor instance (e.g. Mark or John).

Two types of links are used to connect node instances (precedence and assignment links). Both are visualised as arrow between two nodes. Figure 4-5 shows an example of a procedural model in D2P which continues the scenario started in Chapter 3. Further screenshots of procedural models in D2P can be viewed in Appendix I.

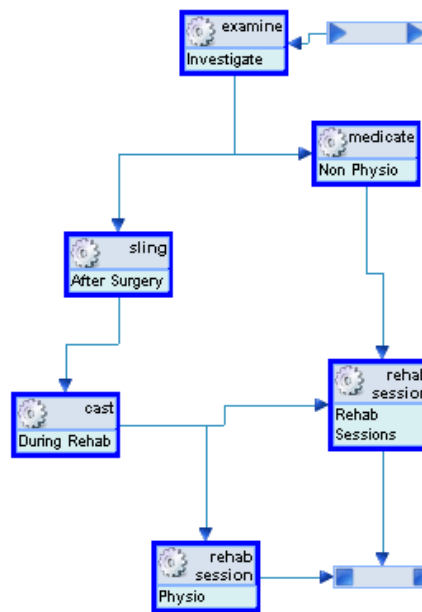





















Figure 4-5 - Example Procedural Model in D2P

The pallet symbol and node design of each the seven declarative nodes (Serial, Fork, Include, Exclude, Precede, Succeed and Task Type) are shown in Table 4-3. The user would click the pallet symbol to create a node on the graphical modelling surface.

Table 4-3 – Declarative Node Summary

| Node Type Name | Pallet Symbol   | Node  | Mode Symbol<br>(On pop-up information view)   |   |   |   |
|----------------|---|---|---|---|---|---|
| Task Type      |  |          | -   |   |   |   |
| Serial         |  | e.g.<br> | -   |   |   |   |
| Fork           |  |   | -   |   |   |   |
| Include        |  |   | <br>Weak | <br>Strong |   |   |
| Exclude        |  |   | -   |   |   |   |
| Precede        |  |   | <br>Weak | <br>Strong  | <br>Weak Chain | <br>Strong Chain |
| Succeed        |  |   | <br>Weak | <br>Strong  | <br>Weak Chain | <br>Strong Chain |

The node type icon in the upper left panel of each node serves as the drag handler. Right clicking on the bottom panel of the node will show the menu for the node. Right clicking



on the top right panel of the node will show the pop-up detailed information dialog for that node.

For each node, the upper right panel displays the name of the node type and the bottom panel displays the name of the node instance. For example the node type may be ‘Task’ and the instance name may be ‘Examine’ or the node type may be ‘Serial’ and the name ‘SE12.1’.

Two types of links are used to connect node instances (trigger of and target). Both are visualised as arrow between two nodes. Figure 4-6 shows an example of a declarative model in D2P. Further screenshots of declarative models in D2P can be viewed in Appendix I.

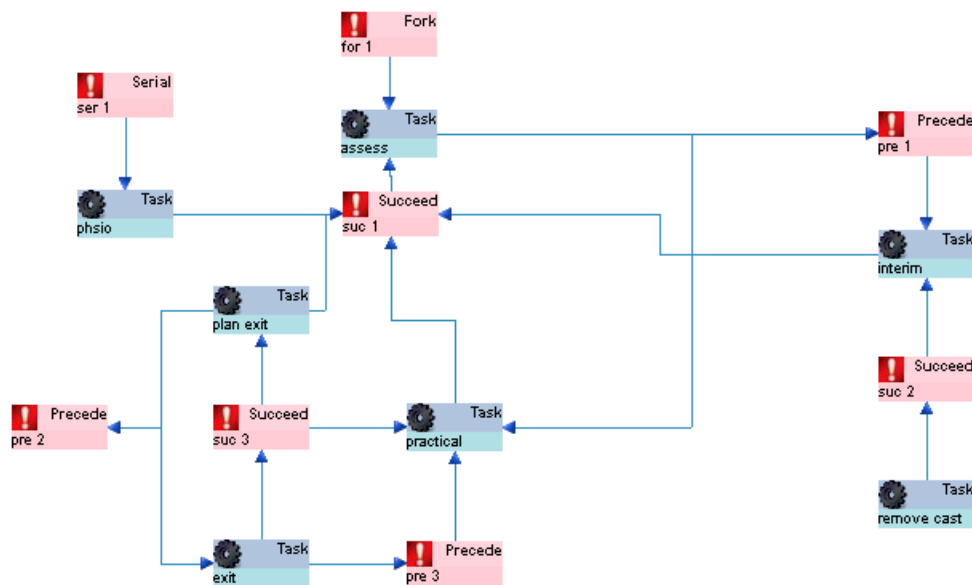


Figure 4-6 - Example Declarative Model in D2P

Detailed information regarding each node in the declarative model is available as a pop-up view by clicking on the node type display. The information display for the task type nodes consists of the current cardinality value for that task type and the current state. The information display for each constraint type consists of the current constraint attribute values, the triggers and targets of the constraint and the current state of the constraint as shown in Figure 4-7.

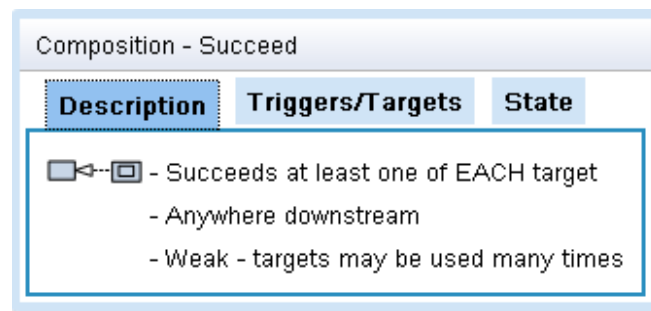


Figure 4-7 - Pop-Up Information View for Constraint Nodes

### Links and Linking Rules

Each kind of relationship (precedence, assignment, trigger and target) is represented as an arrow between two nodes, as seen in the examples given above. The linking rules described in Section 3.4.2 and 3.4.3 have been embedded into the system to allow for computational support in modelling building. This support manifests in automated relationship checking (i.e. illegal connection will be prevented by the system). On a link being requested between two nodes, the list of link rules is checked. If the link cannot be made an error message will be displayed to the user and the link will not be added to the model.

### Model Editing: Step Through

Creating both procedural and declarative models follows a similar process.

Create node:

1. Select the required node type from the node pallet menu e.g. click the task node icon in the node menu of the procedural pallet.
2. Enter the required details on the pop-up dialog to instantiate the node. The various nodes require different information e.g. the task node requires a task type (this can be selected from a drop down list which is populated by any predefined task types e.g. 'physio session'), an instance name (e.g. 'session 2') and join logic (and/or join).
3. The newly created node instance will appear on the work surface. Once the node instance appears on the work surface it can be moved (through drag and drop), removed (through a menu option) and its attribute values can be modified (through a menu option.)

Create link:

1. Select a node to serve as the source of the link by selecting the 'set as source' option from the nodes menu.

2. Select a node to serve as the target of the link by selecting the ‘set as target’ option from the node menu.
3. If the link is allowed (linking rules will be checked by the system to prevent unlawful links such as a constraint node to a constraint node), an arrow will appear on the work surface connecting the source node to the target node. Once a link is established it can be removed (through a menu option accessed by hovering over the link).

### **Model Editor Primitives**

D2P enables the following atomic user actions:







- *Create node (click node type in node menu of either procedural ore declarative pallet)*
- *Delete node (menu option on any node)*
- *Set node as source (menu option on any node)*
- *Set node as target (menu option on any node)*
- *Set node attributes (menu option on any node)*

### **4.5.3 Verification**

Computational analysis tools can provide multiple and alternate views on core process structure to emphasise certain aspects of that structure in a readily digested form. Alternate views on process structure are often used to identify where problems may be occurring.

The computational support focused on in this work is the verification view. This view essentially assigns a value to each constraint in D depending on the current composition of P. The state of a constraint is volatile; it changes depending on the current state of the procedural model. Due to this behaviour, constraint states are given through an on-demand view. This strategy has the further advantage of not affecting performance (after each model update, the status of each constraint does not have to be re-calculated.)

The verification tool can be launched at any time as one of the views available in view menu or as a menu operation available on each node. The verification view, as pictured in Figure 4-8, gives an overview of the entire document. Alternatively, as already described, quick access to the individual states of the constraints is available via the information display on individual nodes in the declarative model.

| Verification View |             |   |                      |
|-------------------|-------------|---|----------------------|
| Cardinality       | Serial/Fork | Include/Exclude   | Precede/Succeed      |
| Resource          |             |   |                      |
| Name              | Count       | State   |                      |
| examine           | 1           |  | Fulfilled if enacted |
| cast              | 1           |  | Fulfilled            |
| remove cast       | 1           |  | Fulfilled            |
| rehab session     | 1           |  | Fulfilled            |
| medicate          | 1           |  | Fulfilled            |
| sling             | 0           |  | Fulfilled            |

Close

Figure 4-8 - Screenshot of the Verification View

The verification view is split into four tabs. Each tab contains all the instances in the declarative model belonging to a specific constraint family. Basic information is given for each constraint (its name, triggers and targets). The current state of the constraint is visualised as a colour. The five possible states of a constraint described in Section 3.4.5 have each been assigned a colour; well formed/enacted – green, well formed/ongoing-amber, well formed/not triggered – black, malformed/temporary – pink, malformed/permanent – red.

To generate a state for each constraint a computational navigation through P is made. The pseudo code for the verification algorithms for the serial, fork, include, exclude, precede and succeed constraints is given in Appendix B. The view is used to highlight error in the procedural model. Particular aspects of the model can be targeted depending on the current state of constraints. Correcting error in the model is user-driven.

### Model Editor Primitives

D2P enables the following atomic user actions:

- *Launch individual constraint view (click on the node type name of any declarative node)*
- *Launch general verification view (1/click on the launch verification menu option in the view menu of either the procedural or declarative pallet, 2/click on the launch verification menu option of any node)*

#### 4.5.4 Enactment

Procedural models serve as a guide as to what tasks should be performed and when they should be performed. The enactment mechanism provides further information by giving the current status of tasks involved in the work. The process enactment logic described in Section 3.4 has been embedded into the D2P application. A visualisation scheme has been applied to task enactment logic; blue for waiting, yellow for ready, black for ongoing, green for finished and red for cancelled. To change the state of the task node instance, a ‘set state’ menu operation is available. The dialog restricts the available state changes for that task based on the state of the model and the enactment rules as described in Section 3.4.

It is important to note that D2P does not automatically perform the direct work involved in the process; this is either a physical task performed by the knowledge worker (e.g. performing an operation) or an information task using some other productivity tool (e.g. word processing.)

The procedural model implements a mixed execution control system. The responsibility of transferring tasks through the various states in their life cycle is distributed amongst the user (human-centred process execution) and the system (system-centred process execution). System centred control will trigger state transitions automatically. Human centred control will leverages users to manually trigger a state transition.

##### **System Control**

Links connect tasks to form precedence relationships amongst tasks. Where possible, the enactment engine reasons about task state changes to automatically activate links which then may have impact on the state of any tasks which are the target of that link by the link. System control of state transitions is centred on the waiting to ready transition. When the *input* link of a task is activated (a preceding task has completed or cancelled) and the logic condition of the task is satisfied, if the task is in a waiting state, it will be automatically set to ready. When a cancelled item receives an inflow through its *input*, it is not restarted.

##### **Human Control**

Enactment under control of users:

- If a starter task (part of a collection or a task part of a structured task with input from a start node only) a user determines when to set a task from ready from waiting.

- A user determines when to set a task from ready to ongoing.
- A user determines when to set a task from ongoing to completed.
- A user determines when to set a task from waiting, ready or ongoing to cancelled.

If P contains no precedence links (i.e. P is a collection), the enactment engine offers little active support. In this case, the users have full control of the process, and they have to do all the work of task enactment manually.

### **Impact on Model Editing**

Enactment has an impact on the editing of procedural content. For example:

- *Removing Content* - P forms a permanent record of what work took place. Therefore no model change should remove record of what has occurred.
- *Adding Content* - P forms an accurate record of what work took place. Therefore users should be prevented from creating a false record of what occurred.

Appendix D summarises the editing rules implemented to uphold these requirements.

### **Model Editor Primitives**

D2P enables the following atomic user actions:

- *Set to ongoing (via set state menu option of task node).*
- *Set to completed (via set state menu option of task node).*
- *Set to cancelled (via set state menu option of task node).*

## **4.5.6 Individual and Group Sessions in D2P**

The figure below gives an example timeline for a room. Some blocks in Figure 4-9 represent when the room is dormant (no active users). A session begins when a user logs into a room. A group session is when multiple people are logged into the same room at the same time. The workgroup then consists of the people logged into the session and are listed in the user list. Each room can have multiple sessions performed serially. In PowerMeeting, synchronous collaboration occurs when multiple people are working in the same room. People can join sessions at any time by logging into the relevant room.

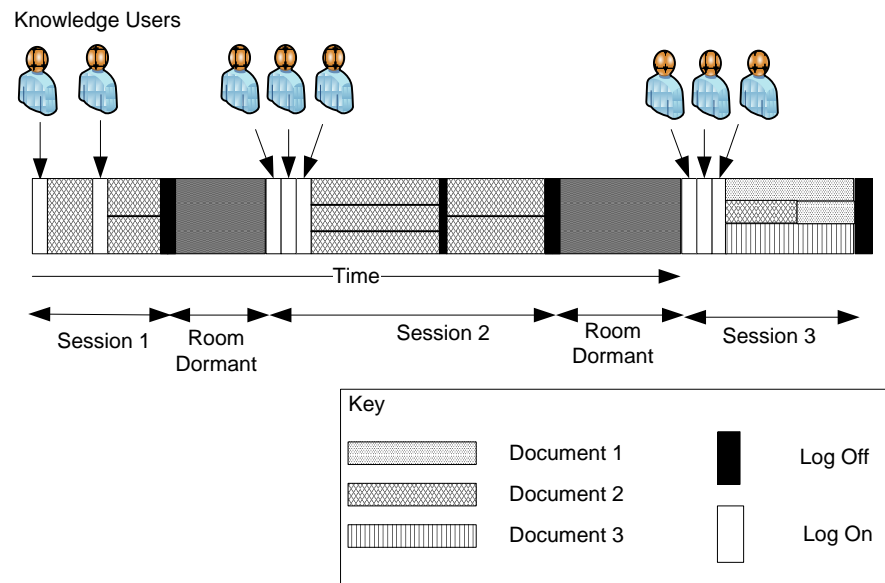


Figure 4-9 - Session Arrangement over Time

The diagram portrays both tightly and loosely coupled synchronous work. Session 2 begins by three users logging into the session at the same time and working in tight synchronous fashion where each user has the same view i.e. browse and modify documents together. The document then becomes a shared workspace which participants work on together with shared control as the graphical content is kept in sync across all connected clients; users can see actions as being performed in real-time. To continue the healthcare scenario, a tight synchronous session may involve users from different organisations, such as a Bone specialist, an orthopaedic doctor and a physio-therapist to develop the treatment program for an individual patient.

The tight integration of communication and shared modelling spaces allows the interplay of process-based “structured” work with collaborative “un-structured” work such as chatting. Typically, additional textual/audio communication channel and/or a group pointing device are used to coordinate their joint actions and facilitate tightly coupled collaboration. Such a shared workspace with fast interaction and feedback is used to establish and develop a shared and common understanding. For example, the various opinions of preferred model composition can be expressed and a consensus reached for a final agreed upon model.

Collaboration can also be less tightly coupled. Multiple users may be logged into the same room, but users can choose to work independently. Participants share the same set of artefacts and have the group awareness, however, they can have separated focus of

attention (work on different documents which exist in the room) to allow users work alone when they wish to do so. They are now still in the same room but can browse and modify content independently. For example, multiple healthcare workers maybe logged into the same room but work on the documents of different patients. This is shown in Session 3 in Figure 4-9.

The figure also shows how group working sessions can be planned or spontaneous. Session 1 begins with an individual working alone. During that session, that individual invites another worker to join and work in the same session. A spontaneous group working session may be triggered when a solitary user requires the opinion of or agreement from another team member for example.

On the other hand, session 2 and 3 is a planned session where multiple workers have organised in advance to meet at the same time and work in the same session. A planned meeting is likely to have a formal agenda and agreed participants. Planned meetings could be on a regular basis, for example, senior doctors may analyse and discuss the progress of patients on a weekly basis.

## 4.6 D2P Application Development

With a domain-oriented approach to the meta-model used to underpin a way of working, it is important that the meta-model is readily created/adapted to suit the domain of its application. The life cycle of a meta-model (see Figure 3-2) is complicated with many sources of influence. Language needs emerge as understanding of the domain develops.

This work has focused on how the life cycle of instance and type models can be supported and controlled. It is equally important to control and facilitate the life cycle of meta-models. It has been shown how cooperative hypermedia environments, similar to PowerMeeting, can be used to rapidly build support for emerging information structures (Wang 2008). The work advocates a process of meta-model discovery which from informal to formal with main steps of:

1. Begin with wholly informal modelling.
2. As structure and behaviour becomes clear, certain useful concepts and links become candidates for development of formalised support.



3. Modelling continues with until deficit is in language is identified. Return to step one.

This was the process followed by this research effort. As understanding of the problem developed, a formal meta-model gradually emerged and was implemented (resulting in the D2P tool).

The PowerMeeting framework can be used along with this methodology to create and adapt applications. Key to this is the use of GWT which allows developers to write their front end in the Java programming language using graphical interface libraries (providing a rich set of interface widgets) similar to Java Swing. The GWT compiler then converts the Java classes to browser compliant JavaScript and HTML.

A plug-in is created by sub classing the SharedModel and its corresponding SharedGadget class of the PowerMeeting framework to create its GUI component. The user interface and underlying application logic are developed in Java using any Java Integrated Development Environment. Eclipse was used in this research effort. As indicated in architecture diagrams, accommodating language requirements is limited to professional users only (although the high level technical protocols described in this chapter ensure that professional users only need a basic grip of Java instead of several Web technologies to be able to make an accommodation).

During development the plug-in (D2P) was debugged, unit tested and then system tested in production mode. Testing was an important part of implementation process to ensure the internal consistencies, correctness and reliability of the artefact developed.

## 4.7 Flexibility through Tool Choice

Much of the work surveyed in Chapter 2 was ‘system-centric’, meaning that work processes are straight-jacketed by the fixed support prescribed and provided by the PAIS, as oppose to the system accommodating the way work is enacted in reality. The variety and richness of application domain is such that the development of a single tool fulfilling all requirements is difficult, perhaps impossible.

As identified, a modern approach to addressing the diverse needs of work support is component-driven. Each component supplies a different way of working. The choice of component which is used for the work is left to the user or maybe rule-driven. Systems realising this way of working are becoming very important. The PowerMeeting ‘plug-in’ approach is aligned with this way of working. The ready extensibility of the environment addresses the need for different tools to provide different types of support. This is in alignment with the dynamic capability best practises presented in (Daniel and Wilson 2003). Different groupware applications residing in an integrated system can be launched to support the task at hand.

The ready extensibility of the environment addresses the need for different tools to provide different types of support. For example, integrated support may include a whiteboard for very informal group collaborations, a voting tool to understand group opinion on issues and a rich text editor for group document development. To further the medical based scenario, the health clinic may develop an application for members of the public to submit health related questions and chat in real-time to medical professionals (the application still leveraging PowerMeeting for a coherent system).

## 4.8 Approach Summary

The integrated framework presented in Chapter 3 comprises of a conceptualisation plus lower level realising technologies. Table 4-4 summarises the approach through key design dimensions. Each dimension of design has a treatment which contributes to addressing a critical area described in Chapter 2.

Table 4-4 – Overall Approach Description

| <b>Design Dimension</b>                      | <b>Treatment</b>                     | <b>Critical Area</b> |
|--|--------------------------------------|----------------------|
| <i>Specificity (planned, agile, chaotic)</i> | Agile                                | 1                    |
| <i>Mechanism</i>                             | Under specification                  | 1                    |
| <i>Language</i>                              | Hybrid (procedural and declarative)  | 1                    |
| <i>Model</i>                                 | Graphical                            | 1                    |
| <i>Grain</i>                                 | Operational                          | 1                    |
| <i>Control (human v computer)</i>            | Mixed                                | 1                    |
| <i>Control (central v localised)</i>         | Mixed                                | 1                    |
| <i>Work mode</i>                             | Mixed - individual and collaborative | 2                    |
| <i>Collaborative work mode</i>               | Mixed – synchronous and asynchronous | 2                    |
| <i>Development paradigm</i>                  | Component-driven/SOA                 | 3                    |
| <i>Delivery</i>                              | Web-based                            | 4                    |

## 4.9 Conclusion

The low level technical protocols which have been used to implement the conceptualisation presented in Chapter 3 have been described. The primary function of the implementation was to realise the techniques and way of working of the conceptual design in a working software system. Requirements of different user types were identified which further constrained the form of the implementation.

The Web-based PowerMeeting framework was leveraged. PowerMeeting provided a framework for rapid Web-based collaborative application development. A RIA called D2P was developed and integrated into the PowerMeeting framework. D2P is primarily a hybrid model editor which allows procedural and declarative content to be created, visualised, analysed, verified and executed. The chapter closed with brief description of how PowerMeeting can be used to component-driven process support.

# Chapter 5

---

## Evaluation: A Usability Study

A Design Science (Hevner, March et al. 2004) research approach has been adopted in this research project. Evaluation is a crucial stage of the Design Science (Hevner, March et al. 2004) research approach. To understand the value of any artefacts created, their strengths and weaknesses must be extracted. This is underlying motive of the evaluative effort.

This chapter describes a systematic and rigorous evaluation effort. Key metrics are identified and the supporting evidence for each metric is given. The majority of this chapter is concerned with the design and execution of an empirical usability study. Results are reported which demonstrate the usability of the hybrid modelling concepts and the D2P tool which implements them.

### 5.1 The Purpose and Scope of Evaluation

Before proceeding further, it is important to understand the target of evaluation effort. Evaluation efforts in this research project either target the concept (presented in Chapter 3) or the tool (the D2P plug-in presented in Chapter 4.) In subsequent discussion of evaluation effort, it will be clearly noted what the target of evaluation is.

To appreciate the value of the artefacts of the artefacts generated through Design Science strengths and weaknesses must be demonstrated through the measurement of relevant metrics. The general form of evaluation is summarised in Table 5-1. Evaluation efforts generally establish or observe the effect that a *treatment* of a *factor* has on a *metric* through its *measurement* via a *data collection technique*. What is observed is then compared to any prior *predictions* made (Zelkowitz and Wallace 1998).

A pragmatic approach has guided the evaluation process. Whilst a strong effort has been made to ensure conformity to recommended practise, an underlying principle has been not whether the evaluation performed followed recommended practice in minute detail, but whether the study and its conclusions taken as a whole represent a credible effort.

Table 5-1 – Components of an Evaluation Effort

| <b>Component</b>       | <b>Definition</b>  | <b>Operationalisation</b>   |
|------------------------|--|---|
| <i>Factor</i>          | A type of influence on user goals. This influence can be ‘tuned’ to help achieve user goals. | Approach to process support; made up of planning approach and supporting technology. Various approaches are described in Chapter 2.                         |
| <i>Treatment</i>       | A specific value assigned to a factor.   | Dual graphical declarative and procedural models delivered via Web-based cooperative modelling environment. See Chapters 3 and 4 for full description.      |
| <i>Metric</i>          | A quality generated from application of treatment to problem domain.                         | See Section 5.2 for a summary of relevant metrics.  |
| <i>Data collection</i> | How reality will be observed i.e. how the measure is taken.                                  | Each metric has been paired to at least one relevant technique. See Section 5.3 for a summary of potential techniques and justification of usability study. |
| <i>Design</i>          | Particular design for the chosen data collection technique.                                  | See Section 5.5 for a summary of the usability study design.  |
| <i>Results</i>         | The resultant behaviour caused by application of the treatment.                              | For a summary of the various results and evidence see Section 5.6.  |

For this research project, classification of evaluative effort as summative (evaluation results do not feed back into artefact development) or formative (evaluation results feed back into development of artefact) is difficult. During artefact development, each metric was considered less formally/intuitively in a formative way by constant researcher-led mini evaluations. The usability study described in the remainder of this chapter forms the summative evaluation effort.

## 5.2 Key Metrics

It should be demonstrated by application to the problem domain that the artefact leads to a measurable improvement in some relevant metric. The key relevant metric in terms of solving the business problem described in Chapter 1 is effectiveness; does artefact effectively address the identified business problem? Measurable metrics are required to assess artefacts generated. This vague attribute will be operationalised by drawing upon the revised Technology Acceptance Model (Szajna 1996) which attempts to explain and predict why users sometimes accept and sometimes reject information systems. It relies on central notions of usefulness and usability:

- Usability - extent to which intended users use the artefact?
- Usefulness - extent to which an application contributes to the enhancement of the user’s performance and help a people accomplish their task more effectively.

Chapters 3 and 4 developed a running scenario which is positioned as evidence of usefulness. This chapter focuses on evaluating the usability of the D2P tool. Importance is given to usability as of the great weight attached to it by an interactive modelling approach.

## 5.3 Data Collection Techniques

With data collection techniques, we begin to move from an attribute of interest to a way of generating an independent source of data which provides evidence for that attribute. Data collection techniques are then ways of sampling phenomena. They must be well-executed to ensure the reliability of results. The reliability of measurements can be improved by reducing the impact of researcher intuition by using accepted data collection techniques.

This section focuses on possible techniques for pairing to the usability metric. This is a problematic metric as a myriad of potential empirical and non-empirical, subjective and objective techniques exist which could be used.

The process of pairing the effectiveness metric with an evidence generating technique is vital. Different techniques generate different kinds of evidence. On the basis of which kind of evidence can we make well founded statements about the usability of the approach? There are no generally accepted answers to this question in the domain of BPM. The merits of several potential techniques are therefore considered.

A key classification of data collection technique is by its engagement with real world setting. Although the data collection techniques below could be represented on a spectrum, purely for simplicity, each technique is classified as either distant or close in terms of engagement with the real world.

The most prominent close and distant techniques are summarised in Table 5-2a and Table 5-2b. It should be noted that the techniques are drawn mostly from the Quantitative, Positivist Research paradigm with exception being case study. This is due to the reliance of Design Science (Hevner, March et al. 2004) on techniques ‘borrowed’ from this paradigm. The below techniques and descriptions have drawn on several texts geared to the coverage of IT evaluation; (Dix, Finley et al. 2004), (Zelkowitz and Wallace 1998) and (Siau and Rossi 1998).

## Close

In more mature fields such as the physical sciences, it is often necessary that rigorous empirical research be undertaken to evaluate the effectiveness of an artefact. To move up the evaluation hierarchy, the validity of any claims of effectiveness need to be tested empirically in situation more closely approximating practice rather than justified by logical argument alone. Empirical techniques employ varying forms of user participation either in controlled or real world environments.

## Distant

These are non-empirical techniques which are removed from real world setting (do not involve end users). Distant techniques abstract over the many complexities of actual human-computer interaction. Distant techniques are often quicker and cheaper. Therefore they are often used as preliminary evaluations to build confidence before more intensive and expensive user based studies are performed.

Table 5-2a – Data Generating Techniques (Close)

| Technique                   | Description   |
|-----------------------------|---|
| <i>Experiment</i>           | <p>A controlled laboratory experiment is the study of an artefact in a controlled environment. The controlled environment is then an abstract approximate of the problem domain we are interested in and assesses phenomenon only within that abstracted domain. Users are taken out of normal environment. The controlled environment is artificially constructed so independent variables can be manipulated, "noise" variables are controlled or accounted for, and the resulting impact on the dependent variables is observed. A control group serves as a base. Experimental groups are the same as control groups except independent variables are adjusted. The control group is necessary to ensure that is the independent variable which causes the result.</p> <p>A usability study is taken to have similar properties to an experiment except the control group is missing.</p> |
| <i>Field Study</i>          | <p>Field study is an experiment or observation which takes place in the organisational environment. The researcher considers the artefact in use in a natural work situation. With a field experiment, the independent variable is not manipulated, but large amounts of data are collected in order to determine its effect. The dependent variables are systematically measured i.e. an experimental design framework is used, but experimental control is absent.</p>  |
| <i>Case study</i>           | <p>An artefact is studied artefact in depth over a long period in a business environment. Thus, like field study, the case study again examines a phenomenon in its natural setting, employing multiple methods of data rich collection and analysis. The boundaries of the phenomenon are not clearly evident at the outset of the research and no experimental control or manipulations are used.</p>   |
| <i>Query</i>                | <p>The user is queried directly through to gain attitudes, opinions, impressions and beliefs through different styles of interviews, questionnaires and surveys.</p>  |
| <i>Heuristic evaluation</i> | <p>Guidelines or principles are used to critique or guide the design of a prototype. The most famous set of guidelines is Nielsen's 10 usability heuristics (Nielsen 1994). Users would independently assess the artefact for conformity to the guidelines.</p>   |

Table 5-2b – Data Generating Techniques (Distant)

| Technique                               | Description - the kind of evidence each technique can generate and how it is generated  |
|---|---|
| <i>Cognitive walkthrough</i>            | A detailed review is undertaken of the sequence of actions the interface of some artefact requires a user to accomplish a task. Experts step through the sequence and attempt to identify possible usability problems and assess how easy it is to learn. Multiple experts work independently and often use standardised evaluation forms.  |
| <i>Heuristic evaluation</i>             | Guidelines or principles are used to critique or guide the design of a prototype. The most famous set of guidelines is Nielsen's 10 usability heuristics (Nielsen 1994). Experts would independently assess the artefact for conformity to the guidelines.  |
| <i>Simulation</i>                       | An experiment is performed (as above) except input is from simulation data rather than from human participants. In other words, the artefact is loaded with artificial data.  |
| <i>Model-based evaluation</i>           | A reference model is used to provide a framework for structured analysis of the tool. For example; <ul style="list-style-type: none"> <li>• The SEQUAL (Krogstie, Sindre et al. 2006) model quality framework could be used to assess how far the tool can facilitate the various model qualities identified by SEQUAL.</li> <li>• The workflow pattern initiative (Aalst and Hofstede 2003) could be used to assess the expressiveness of the language.</li> </ul> |
| <i>Scenario</i>                         | Construct detailed scenarios around the artefact to demonstrate its utility through when and how the method may be used.  |
| <i>Comparison with previous studies</i> | Leverage the wealth of past research by using it to form benchmarks against which various qualities of the artefact can be judged.  |

Often the quality we seek to measure cannot be measured absolutely. We can get deeper insights into it by studying usability by using multiple and complimentary data collection techniques. A usability study with human subjects augmented with a query approach has been selected as a viable data generating technique for the usability metric.

## 5.4 Justification of Data Collection Choice

(Perry, Porter et al. 2000) highlights the importance of experiments in software engineering and describes experiment results as powerful additions to research papers. Experiments in the conceptual modelling domain have been widely reported, for example, (Agarwal, De et al. 1999) use experiment to compare procedural and object-oriented modelling languages and (Cardoso 2006) use experiment to validate a control-flow complexity metric. In process-oriented research domains, (Weber, Reijers et al. 2009) have used experiment to assess the impact of varying the amount of declarative information (constraints) on the user's ability to generate a solution. Recently, (Fahland, Lubke et al. 2009) has called for



more empirical investigation on the relative strengths and weaknesses of declarative and procedural modelling languages.

Both field and case study are discounted for similar reasons. Expensive field work in terms of time, cost, organisation and analysis cannot be justified given the scope of research goals. The area of interest can be divorced from its real-world setting via well designed and executed usability study. In other words, the richness of the real world environment provided by these techniques is not required by the stated research goals. The key criticism of the experimentation approach is the lack of richness found in real world settings such as multi-tasking and work interruptions. Thus the key criticism is of reduced significance in this study.

The case study is geared to “real users using real system to solve real problems”. The nature of the prototype developed falls short of an industrial strength system which can be deployed into a real world environment for a case study.

Heuristic and cognitive walkthrough are quick, easy and low cost. They were discounted to leverage a functioning software prototype to a greater extent by involving human subjects to provide more informative results. Such distant approaches are subjective and non-repeatable. Whilst these techniques are very useful both during and after design, the approach of experimentation was favoured due to its objectivity and scope for repetition in testing low level hypothesis.

Model-based analysis to examine the expressiveness of the technique was discounted. Many approaches for evaluating modelling languages focus almost entirely on the expressiveness where as usability formed the goal of this study. Model-based analysis which focused on the quality of models and how that quality is facilitated was discounted due to the difficulty of an objective operationalisation of the technique.

## 5.5 Usability Study: Design

The International Standards Organisation (ISO 9241-11) defines usability as ‘the extent to which a product can be used by specified users with effectiveness, efficiency and satisfaction.’

Effectiveness and efficiency are taken to be objective metrics. Although this is clearly relevant to user performance given the system is used, subjective usability is more relevant to the users' decision whether or not to use the system in the first place. Therefore, satisfaction is taken to mean subjective (or perceived) effectiveness and efficiency. Perceived usability is hypothesised to be fundamental determinants of user acceptance (reported current usage and predicted future usage) (Davis 1989). This is why it is important that we collect perceived measures.

The design of the usability study described in the following sections gathers data relating to these aspects. The figure below shows an overview of the usability study process.

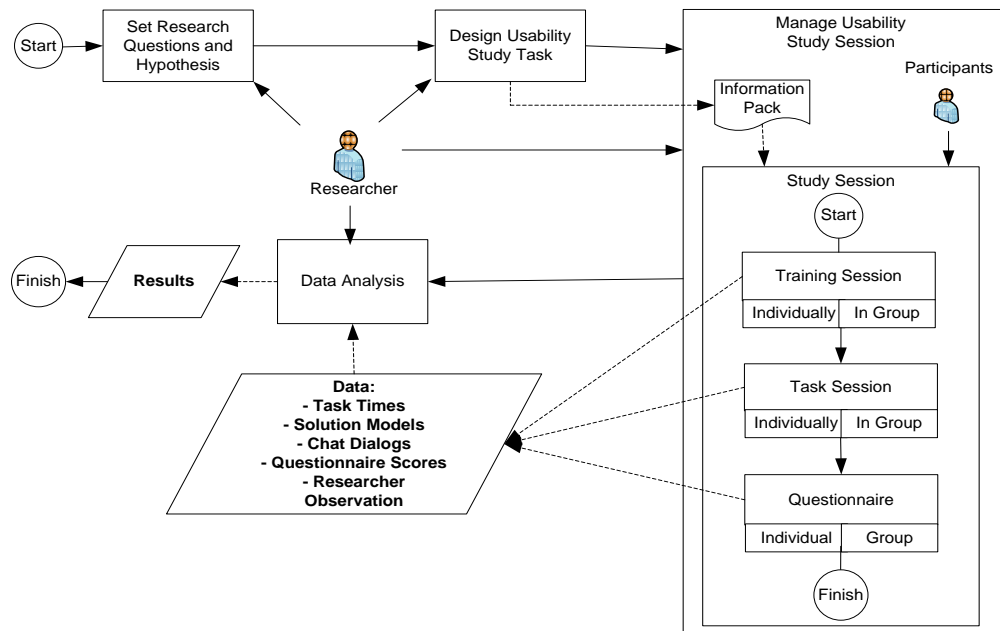


Figure 5-1 - Summary of the Usability Study Process

The study follows an experiment meta-design set out in (Gemino and Wand 2004) and also (Tullis and Albert 2008). Table 5-3 presents a high level summary of the core usability study design. Each component is elaborated upon in subsequent sections.

Table 5-3 – High Level Summary of Usability Study

| Component               | Operationalisation   | Section             |
|-------------------------|--|---------------------|
| <i>Research context</i> | Flexible process support with hybrid interactive models  | 5.5.1               |
| <i>Hypothesis</i>       | Individually and in groups, PowerMeeting and D2P can be activated by users to interpret declarative models and articulate procedural models. | 5.5.2               |
| <i>Study Design</i>     | Using the D2P modelling tool and case data, ask human users to complete a problem solving task.  | 5.5.3<br>&<br>5.5.4 |
| <i>Results</i>          | User experience (through questionnaire), task result (through transcript analysis) and researcher observations.                              | 5.6                 |

### 5.5.1 Research Context

*In this section the problem domain and key pieces of terminology are identified. The relevance and importance of the work is established through a brief summary of the related work and how the study intends to build on this body of work.*

The goal of flexible process support is to help people plan and perform their work processes. The end goal of all research in this field is then contributing in some way to understanding how this can be achieved.

We contribute to this goal by assessing the usability of an approach characterised by user interaction with graphical models to help plan, analyse, improve and enact their work. Contribution to process support is made in two main ways:

- Procedural modelling has dominated the interactive modelling approach to process support. Insight into the impact of declarative modelling techniques is extremely limited. This work generates performance data for the use of a hybrid (procedural and declarative) modelling technique. Furthermore, the impact of working as individuals and working in groups is examined.
- The usability of interactive modelling has been evaluated mostly through detailed scenarios, some field studies and some through model-based analysis. Usability studies are rare. This study then is important as the usefulness of experimentation as evaluation tool for the flexible process support domain is assessed.

The study potentially has impact beyond BPM. Findings may have implications for primarily for the workflow, Web Service composition and conceptual modelling domains as modelling techniques are frequently employed in these fields e.g. for IS development projects.

## 5.5.2 Hypotheses

*In this section we state what we intend to evaluate. This will take the form of research questions and speculative hypothesis that can be objectively supported (or refuted) by resultant usability study data.*

### Research Questions

Procedural interactive models have been successfully used to achieve flexible process support when consumed individually and in groups. Domain experts are often put in charge of the modelling, although they do not necessarily are modelling experts. It is unclear whether end-users are really capable of activating (creating and adjusting a plan) when using a hybrid modelling approach. The impact of the use of group working on a hybrid modelling approach is also unclear.

The original research question asked ‘Is the approach usable by novice and semi-skilled knowledge workers?’ In this usability study we are interested in the impact of a hybrid modelling approach on the success of task planning as performed by individuals and groups:

- Can users, working individually, successfully activate a hybrid interactive approach to agile planning, as enabled by a PowerMeeting and D2P, to plan a valid business process?
- Can users, working in groups, successfully activate a hybrid interactive approach to agile planning, as enabled by PowerMeeting and D2P, to plan a valid business process?

### Hypotheses

1/ Working individually, users can successfully use a hybrid interactive approach to agile planning, as enabled by PowerMeeting and D2P.

2/ Working in groups, users can successfully use a hybrid interactive approach to agile planning, as enabled by PowerMeeting and D2P.

Two components of the hypothesis presented above require further qualification.

- Users are defined as those with a basic familiarity with PAIS. Furthermore, users will possess little domain knowledge (relating to the theme of the problem solving task).
- A successful use matrix is described in Table 5-4.

Table 5-4 – Successful Use Matrix

| <b>Metrics</b>   |                  |  |   |
|------------------|------------------|--|---|
| <b>Usability</b> | <b>Actual</b>    | <b>Effectiveness</b><br>(task success)   | <b>Learning</b> Approach concepts* and how realised in tool understood correctly by users.  |
|                  |                  |  | <b>Performance</b> Approach concepts* and how realised in tool activated correctly by participant.                                    |
|                  |                  | <b>Efficient</b><br>(time on training and task)                                | <b>Learning</b> Acceptable amount of effort required by the user to learn approach concepts* and how realised in tool.                |
|                  |                  |  | <b>Performance</b> Acceptable amount of effort required by the user to activate approach concepts* and their realisation in the tool. |
|                  | <b>Perceived</b> | <b>Effectiveness</b><br>(user perceptions of success)                          | <b>Learning</b> Approach concepts* and how realised in tool understood correctly by users.  |
|                  |                  |  | <b>Performance</b> Approach concepts* and how realised in tool activated correctly by participant.                                    |
|                  |                  | <b>Efficient</b><br>(user perceptions of time and effort on training and task) | <b>Learning</b> Acceptable amount of effort required by the user to learn approach concepts* and how realised in tool.                |
|                  |                  |  | <b>Performance</b> Acceptable amount of effort required by the user to activate approach concepts* and their realisation in the tool. |

\*Approach concept:

- *Create/Adapt* - Articulation of procedural models. To make brand new structure and make modifications/corrections to existing procedural models.
- *Interpret/Analyse* - Interpretation of declarative models. Use of model inspection and automated verification view to check state of individual models and alignment between models.

### Potential Outcomes

Table 5-5 summarises the potential outcomes of the study.

Table 5-5 – Potential Outcomes

| <b>Individual Task - Level of support</b> | <b>Group Task - Level of support</b> | <b>Implication</b>   |
|---|--------------------------------------|--|
| <i>Low</i>                                | <i>Low</i>                           | Language not usable, group work has no impact on usability. (A good design will provide accountability for the cause of the result. Usability study data would not provide any information for hypothesis 2 as the low result can't attributed to quality of language or group support. In this case information questionnaire from the questionnaire is used. |
| <i>High</i>                               | <i>Low</i>                           | Language usable, group work not well supported.  |
| <i>Low</i>                                | <i>High</i>                          | Language unusable. Group work extremely well supported as better understanding of language formed in groups.   |
| <i>High</i>                               | <i>High</i>                          | Language usable and remains usable in group working. This is the hypothesised outcome  |

### 5.5.3 Usability Study Design

*This section describes the process of generating data which can support (or refute) the hypothesis. The researcher manipulates independent variables and measures their effect on the dependent variable. The goal of a study is to collect enough data from a sufficient number of subjects (multiple instances of an observation), all adhering to the same treatment, in order to obtain a statistically significant/valid result on the attribute of concern. Frequently this result will be compared to the result generated by another treatment.*

*The impact on the dependant variable should be caused solely by the independent variable and not the result of some unknown influence/unintended factor (noise). In study design it is important to filter out noise. Should noise impact on the result, our ability to interpret or draw conclusions from the study's data is greatly reduced. For that reason a summary of the potential threats is provided.*

*This section is organised around three dimensions of experiment design; independent variables, dependant variables and data analysis.*

#### **Independent Variable: Grammar and Grammar Delivery**

The specifics of the modelling language (the grammar) and how that language will be consumed (tool used to present the grammar) are described.

*Operationalisation* - An intra-grammar study (within one modelling language) based on a hybrid process modelling notation (procedural and declarative). The language is delivered via a Web-based collaborative graphical modelling environment (PowerMeeting and D2P).

*Threats* - Using a modelling tool which implements a way of working implies that the target of evaluation is the tool rather than purely the underlying concept. It has to be accepted that a pure measure of understanding of the concept is not possible as that understanding is always generated through some filter e.g. the design of the modelling tool, the help documents etc. It is therefore important to acknowledge that claims based on the usability study are not based purely on the concept but how it is realised by the tool. Results then depend to a large extent on the quality of the tool which implements them.

**Independent Variable: Case Information and Nature of the Task**

The specifics of the nature of the task participants will be asked to perform are described.

*Operationalisation* – The user task can be described as problem solving. The problem involves both model interpretation and model creation.

- Model interpretation seeks the faithful interpretation of model; a model viewer has the goal of understanding the domain being represented by interpreting the model.
- Model creation seeks the faithful representation of domain knowledge; the model creator has a goal to communicate his/her understanding by representing it in a model.

Participants are provided with researcher designed, error-free information of a medical-based business domain. A medical based domain was chosen to reduce the impact of advanced domain knowledge of participants. The case data consisted of:

- An information pack detailing task instructions and case information (textual data which describes the specific needs of the case) given to participants can be found in Appendix F.
- Declarative model (capturing constraints). The graphical declarative model delivered in D2P can be viewed in Appendix I. Users are free to move model components (as different spatial arrangements may aid interpretation) but not edit the model.

Recent studies have focused on measuring the understanding of the domain using problem solving tasks that require participants to reason about the domain being represented. A problem solving test asks questions that require participants to infer answers from information they have received. The problem solving questions required participants to use the mental representation they had developed to suggest answers for which information was not directly available.

The problem solving task focuses on interpretation of declarative model and articulation of procedural model. As individuals or in groups, the user/s are required to create a procedural model from the given declarative model and case data. Where the task was performed in groups, it is performed synchronously where each team member was isolated from others.

The measurements that can be applied to model interpretation tasks are less obvious than those in script creation as we seek to measure something intangible - the understanding invoked by a model. Given that the cognitive model cannot be directly observed, we assess the performance of interpretation based on articulation task. It is premised that the more complete and accurate answers (models) participants provide, the higher the level of understanding they have developed. This makes an intangible metric (of understanding) tangible.

To complete the task subjects have to demonstrate their learning and performance in the following areas:

- *Ability to create* - Build a procedural model.
- *Ability to analyse* - Check procedural model for correctness.
- *Ability to interpret* - Interpret meaning of declarative model.

We force interpretation of the constraint model by providing a partial case, which mixes explicit instruction with omissions which the user have to resolve for a correct model. The measurement, therefore, focuses on the “deep” understanding a user generates by viewing a diagram and not the elements of the diagram itself.

Throughout both the training and the task sessions, participants could ask the researcher any questions. Where questions were asked by participants, the assistance given usually described some concepts or aspect of the user interface in more detail until the issue was resolved and the participant was in a position to move forward.

Each user will be asked to fill in a questionnaire addressing user experience and perceptions as individuals or group. The questionnaire was either delivered via a paper form and filled in by subjects or delivered electronically.

The first part of the questionnaire gathers information pertaining to the participant’s profile (a self-assessment on a five point ordinal scale of their experience/knowledge in related subjects.)

Closed questions allowed subjects to rate aspects of their experience. Participants made a self-assessment on a five point ordinal scale of various aspects of their tool usage



experience by agreeing or disagreeing with given statements. Each question can be linked to some aspect of performance.

A final open question allowed users justify and elaborate on responses and raise issues missed by the closed statements. The group version of the questionnaire can be viewed in appendix G.

*Threats* - Various factors may unintentionally impact upon results:

- A model interpretation task is being measured by a model articulation task. Competence with procedural modelling is assumed i.e. poor scores are attributed to misunderstanding the declarative model rather than difficulty in articulating a procedural. It is possible that errors are due to difficulty of using procedural concepts.
- Group work has the potential for individual talent to determine the result. We use a technique where consensus and rationale was required for task success. A superior design would be if control or information was distributed to force participation for task success; one party can only do so much before they require discussion, instruction, information or consultation with another.
- The task consisted of only one business process of a given complexity and without using hierarchical models. The size of the task may be unrepresentative of real world cases and thus may not be generalisable to practise. While using more than one case increases the costs associated with the study, consistent results across two or more cases lead to stronger validity.
- The given model does not provide systematic coverage of the entire hybrid modelling language. Further, what we do is not quite the same as collectively executing a business process; changes are not made to the declarative model and the procedural model is not executed. Changes to the declarative model are outside scope as often in recurring processes they would be stable. The focus of this study is on, once have some notion of constraint, can they be modelled correctly?
- The layout of the given declarative model may impact on understanding. Layout was kept consistent for each user and group.
- An engaging case highly improves willingness of participants to use the system. A medical based case may not be particularly engaging.
- Subjects may not be committed to tasks, may get distracted or fatigued. This may account for weakness in results. The lengths of the sessions were minimised through thorough planning.

- The training period does not replicate intensive training users are given for professional business applications.

### **Independent Variable: User Characteristics and Group Composition**

The characteristics of the person/s participating in the study are described.

*Operationalisation* – In total, 12 participants were selected:

- Age - Variable
- Profile - Students, IT professionals
- Prior case knowledge (i.e. medical practices) - Little
- Training - All subject to same level of training in preparation for the study.

Out of the 12 participants described above, 6 were chosen at random to perform the task in groups. Each of the three groups consisted of two people.

*Threats* - Various factors may unintentionally impact upon results:

- The nature of the sample population used should be representative of the population we wish to generalise results to. We want to generalise findings to a population which is semi skilled in process-oriented concepts. Our participants then in general were semi skilled in process-oriented concepts. However, this does not afford the generalisation of results to all potential users.
- It can be argued that a small sample size was used. However, small sample sizes can still be meaningful. Laboratory tests have shown 4-5 people can find 80 percent of usability issues (Nielson 1994). In addition many prior studies used 3 or 4 groups to successfully identify usability issues in group evaluation (Dix and Finley 2004).
- It can be argued that the small group size (two) is inadequate. Two people may not be enough to reveal group dynamics, for example the chat function with two people is a conversation. How would the chat function hold up with multiple participants?

### **Dependant Variables**

The dependant variables dimension comprises observable outcomes of the modelling task.

*Operationalisation* – Table 5-6 summarises the dependant variables and how insight is gained into how they react.

Table 5-6 – Summary of Data and Metric Collected from that Data

| Source of Data       | Form of Data            | Dependant Variable   |
|----------------------|-------------------------|--|
| <i>Task</i>          | Graphical Models*       | Actual effectiveness of learning and performance                   |
|                      | Log                     | Actual efficiency of learning and performance                      |
|                      | Observations            | Actual effectiveness of learning and performance                   |
| <i>Questionnaire</i> | Filled in questionnaire | Perceived effectiveness and efficiency of learning and performance |

\* *Graphical Models* - Task success is extracted by scoring the models generated by participants. Models are assessed for the presence of certain properties which can be classified as:

- *Explicit* - can be achieved by looking at case data and using procedural concepts only. However, these properties still need to be codified with system, thus demonstrates understanding.
- *Implicit* - to achieve these properties, declarative modelling concepts have to be interpreted.

Table 5-7 – Marking Scheme for Models Generated by Participants

| Property   | For Full Score    | Type     |
|--|-------------------|----------|
| Start node (1)                                     | Connected         | Implicit |
| End Node (1)                                       | Connected         | Implicit |
| Assess nodes (2)                                   | Assess typed      | Explicit |
| Physio node (2)                                    | Physio typed      | Explicit |
| Plan exit node (1)                                 | Plan exit typed   | Implicit |
| Practical node (2)                                 | Practical typed   | Implicit |
| Interim node                                       | Interim typed     | Explicit |
| Remove node  | Remove typed      | Explicit |
| Locate node  | Locate typed      | Explicit |
| Assess nodes in fork                               | Connected         | Implicit |
| Physio nodes in serial                             | Connected         | Implicit |
| Plan exit before locate and after assess           | Chain from assess | Implicit |
| Practical 1 before locate and after assess         | Chain from assess | Implicit |
| Physio sessions after at least one assess node     | Connected         | Explicit |
| Interim after physio                               | Connected         | Explicit |
| Remove cast after interim                          | Connected         | Explicit |
| Exit after remove cast, arrange exit and practical | Connected         | Implicit |
| Practical 2 after exit                             | Connected         | Implicit |

Table 5-8 presents the scoring system used with the marking scheme i.e. each property is above is assigned a score from the table.

Table 5-8 – Scoring System

| Score | Property                              |
|-------|---------------------------------------|
| 1     | Present (No assistance)               |
| 0.8   | Present (Assistance)                  |
| 0.6   | Partial (No assistance)               |
| 0.4   | Partial (Assistance)                  |
| 0     | Absent (Assistance/No assistance)     |
| -1    | Instances of excess                   |
| -1    | Instances of illogical process design |

*Threats* - A single measure for a metric may not give a true picture (as measures don't always correlate). To combat this affect performance and satisfaction metrics are combined to get a broader perspective.

#### 5.5.4 Data Analysis and Generating Results

*Operationalisation* – How are results generated from the raw outputs of the task and questionnaires? Table 5-9 presents how the data output from the study will be analysed to assess the hypothesis.

Table 5-9 – Data Analysis Techniques Summary

| Source                  | Method  |
|-------------------------|---|
| <i>Graphical Models</i> | Actual effectiveness of learning and performance is gleaned from expert rating of the graphical model produced.<br>Actual learning is inferred as a precondition for task success and time on task. Interval data is provided by the marking scheme. Descriptive statistics are used for data analysis.   |
| <i>Observations</i>     | Less formal observational techniques provided a useful process-oriented perspective: <ul style="list-style-type: none"> <li>• A coding scheme with anticipated events/problematic areas and recorded frequency of occurrences was used.</li> <li>• The conversations generated in the chat tool are also analysed providing valuable insight into the usability of the tool.</li> </ul> |
| <i>Log</i>              | The training session time for each participant was taken. The time taken to finish the task for each participant/group was taken. These measures are used to understand the actual efficiency of the task.  |
| <i>Questionnaire</i>    | Analysis of results for perceived efficiency and perceived effectiveness. Interval data is provided by questionnaire. Descriptive statistics are used for data analysis.  |

*Threats* - Researcher bias could potentially impact on the validity of results. An objective checklist removes subjectivity in the marking of graphical models. Confidence intervals are lower due a lower number of participants. This weakness can be alleviated with further tests.

## 5.6 Usability Study: Results

### 5.6.1 Training

Table 5-10 presents a summary of the training sessions.

Table 5-10 – Summary of the Training Session

| Subject Name     | Room Name | Trained                                    | Time (minutes)  |
|------------------|-----------|--|---|
| Bhavik Shah      | T1        | Group(local)                               | 60  |
| Hilary Danclu    | T2        | Group(local)                               | Due to gap between training and task a further 10 was taken before the task session       |
| Vivek Thomas     | T3        | Group(local)                               |   |
| Zhaupu Wang      | T4        | Group(local)                               |   |
| Weigang Wang     | T5        | Group(local)                               |   |
| Shahzad Quraishi | T6        | Individual (local)<br>(performed as group) | 45<br>Due to gap between training and task a further 10 was taken before the task session |
| Yicheng Zhou     | T7        | Group (local) (performed as individual)    | 60<br>Due to gap between training and task a further 20 was taken before the task session |
| Joe McMahon      | T8        | Individual (local)                         | 30  |
| Graham Cherry    | T9        | Individual (local)                         | 60  |
| Chris Cherry     | T10       | Individual (distance)                      | 50  |
| Michael Edge     | T11       | Individual (local)                         | 30  |
| Su Li            | T13       | Individual (local)                         | 55  |

Mean: 59.1

Standard Deviation: 16.1

#### Room name

The room name refers to the room name used in the PowerMeeting system.

#### Trained

Participants could potentially have been trained in four different ways described below:

*Group/Local* - Multiple participants were trained at the same time and at the same place by the researcher. Participants were not separated for this session. Six participants were trained in this manner.

1. All participants were given a brief introduction to the theme of the work with a 5 minute PowerPoint presentation.
2. Each participant was given a information pack containing:
  - a) Room allocation.

- b) Training session description, training session instructions, training session scenario and training quiz and training answers
  - c) Study session description, study session instructions, study session scenario data and case data
  - d) Questionnaire
3. Each participant used a PC. Subjects were directed to the PowerMeeting site (using a browser of their choosing).
  4. Participants logged on to allocated room.
  5. Participants navigated to and opened an existing D2P document.
  6. Participants were encouraged to read through both help manuals.
  7. Participant's instantiated the training template.
  8. Participants were encouraged to work through training instructions.

*Group/Distance* - Multiple participants were trained at the same time and at the same place by the researcher using PowerMeeting.

*Individual/Local* - A one on one session with researcher and participant working together at the same time in the same place in a group laboratory. The process was equivalent to group/local training. Five participants were trained in this manner.

*Individual/Distance* - A one on one session with researcher and participant working together at the same time in different places using PowerMeeting. One participant was trained in this manner.

1. The presentation and information pack (see above) were emailed to participant.
2. Participant was asked to look through the materials prior to the training session.
3. At a pre-designated, agreed time, researcher and participant logged onto PowerMeeting Participants in the allocated room (using a browser of their choosing).
4. As the participant already had access to materials, the first few minutes were an informal question and answer session using the chat tool.
5. Participants navigated to and opened an existing D2P document.
6. Participant was encouraged to read through both help manuals.
7. Participant instantiated the training template.
8. Participant was encouraged to work through training instructions.

Throughout any type of session, participants could ask the researcher any questions.

### **Example Training Models**

Participants managed to create a valid procedural model with varying degrees of aid. Several screenshots of models created during training are given in Appendix I.

## **5.6.2 Task**

The following list gives the task process for individuals:

1. Each participant used a PC.
2. If necessary, subjects were directed to the PowerMeeting site (using a browser of their choosing).
3. If necessary, participants logged on to allocated room.
4. Participant's instantiated and opened the D2P task template.
5. Using the D2P tool, participants were encouraged to follow study instructions given in information pack.

The following list gives the task process for groups:

1. Participants were split randomly into groups of two. Each team was allocated a room to work in PowerMeeting. Participants sat on opposite sides of a computer lab and were instructed that communication outside of the PowerMeeting system was not allowed.
2. Each participant used a PC. Subjects were directed to the PowerMeeting site (using a browser of their choosing).
3. One member of each team instantiated and opened the D2P task template.
4. Participants were instructed to read through instructions.
5. The team could begin when ready. Using the D2P tool, participants were encouraged to follow study instructions given in information pack.

## **5.6.3 Task: Task Success (Actual Effectiveness)**

Table 5-11 presents the task success results from the participants who performed the task as individuals. The complete breakdown of results can be found in Appendix J.

Table 5-11 – Task Success for Individuals

| Room | Score   | Percentage |
|------|---------|------------|
| 7    | 14.2/18 | 79         |
| 8    | 14/18   | 78         |
| 9    | 11/18   | 61         |
| 10   | 15.2/18 | 85         |
| 11   | 16.8/18 | 93         |
| 13   | 11/18   | 61         |

Mean: 13.7 (out of a maximum of 18)

Standard Deviation: 2.3

The mean score for participants performing as individuals was 13.7 out of a possible maximum of 18. This figure demonstrates that on average a participant with brief training period could successfully complete 76% of the given modelling task. As an example, the model produced by room 8 is given in Appendix I.

Table 5-12 presents the task success results from the participants who performed the task in groups. The complete breakdown of results can be found in Appendix J.

Table 5-12 – Task Success for Groups

| Room | Score   | Percentage |
|------|---------|------------|
| 1    | 12.4/18 | 69         |
| 2    | 13.2/18 | 73         |
| 3    | 16.2/18 | 90         |

Mean: 13.9

Standard Deviation: 2.0

The mean score for groups was 13.9 out of a possible maximum of 18. This figure demonstrates that on average a participant with brief training period could successfully complete 77% of the given modelling task. As an example, the model produced by room 2 is given in Appendix I.

The figure below shows a graphical representation of task success for both individuals and groups. To create this graphical representation it was necessary to allocate names to scoring areas; poor 0-49, average 50-59, good 60-69, very good 70-79, extremely good 80-89 and excellent 90-100.



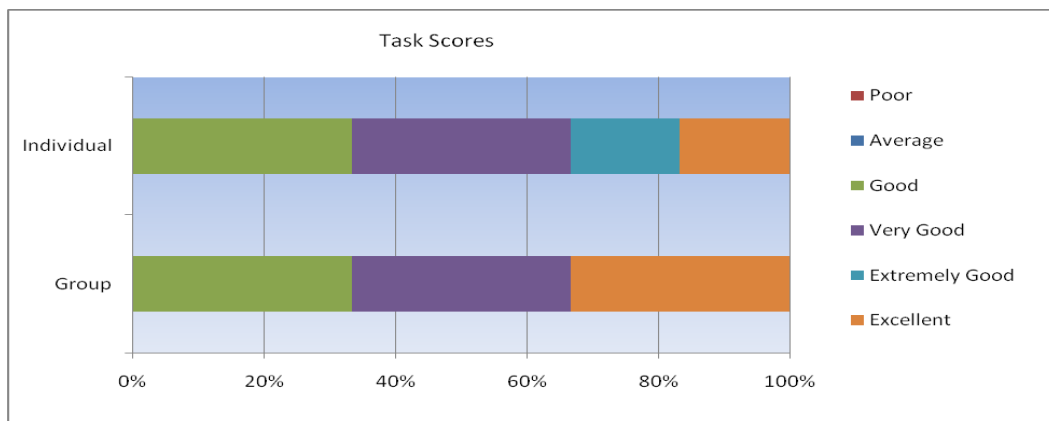


Figure 5-2 - Group and Individual Task Success Results

#### 5.6.4 Task: Task Time (Actual Efficiency)

Table 5-13 presents the task time results from the participants who performed the task as individuals.

Table 5-13 – Individual Task Time Results

| Name                 | Room | Total Time (minutes) |
|----------------------|------|----------------------|
| <i>Graham Cherry</i> | T9   | 45                   |
| <i>Chris Cherry</i>  | T10  | 40                   |
| <i>Joe McMahon</i>   | T8   | 25                   |
| <i>Mike Edge</i>     | T11  | 30                   |
| <i>Yicheng Zhou</i>  | T7   | 30                   |
| <i>Su Li</i>         | T13  | 15                   |

Mean: 31

Standard Deviation: 10.7

The mean measure was 31. This means the average time taken to complete the task was 31 minutes.

Table 5-14 presents the task time results from the participants who performed the task as groups.

Table 5-14 – Group Task Time Results

| Groups   | Room | Total Time (minutes) |
|--|------|----------------------|
| <i>1 Vivek Thomas</i><br><i>2 Shahzad Quraishi</i> | T1   | 40                   |
| <i>1 Hilary Danclu</i><br><i>2 Weigang Wang</i>    | T2   | 30                   |
| <i>1 Bhavik Shah</i><br><i>2 Zhaupu Wang</i>       | T3   | 25                   |

Mean: 32

Standard Deviation: 7.6

The mean measure was 32. This means the average time taken to complete the task in groups was 32 minutes.

The figure below shows a graphical representation of task speed for both individuals and groups. To create this graphical representation it was necessary to allocate names to speed areas; rapid 0-15, very fast 16-30, fast 31-45, satisfactory 46-60 and slow 61+.

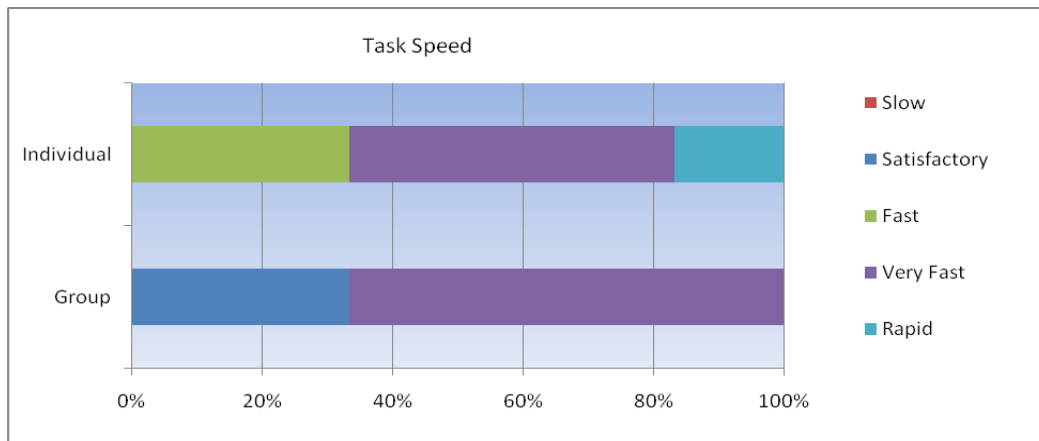


Figure 5-3 - Individual and Group Task Time Results

### 5.6.5 Questionnaire (Perceived Effectiveness and Efficiency)

#### User Profile

Participants were asked to rate themselves on a scale of 1-5 (5 being the strongest) in the following areas:

- Q1 - theoretical knowledge of business process modelling in any context
- Q2 - practical experience of business process modelling in any context
- Q3 – practical experience of process aware tools in any context

Each of the three profile questions was given the same weight in calculation of a total figure. A single figure can be used to classify individuals on ability: 0-5 novice, intermediate 6-10 and expert 11-15.

Table 5-15 shows a summary of user profile data for individuals who performed as individuals.

Table 5-15 – Individuals Profile Summary

| Participant | Q1 | Q2 | Q3 | User Score (Max 15) |
|-------------|----|----|----|---------------------|
| T7          | 2  | 2  | 4  | 8                   |
| T8          | 1  | 1  | 2  | 4                   |
| T9          | 2  | 2  | 3  | 7                   |
| T10         | 3  | 3  | 3  | 9                   |
| T11         | 4  | 3  | 4  | 11                  |
| T13         | 4  | 3  | 4  | 11                  |

Mean: 8.3

Standard Deviation: 2.7

Table 5-16 shows a summary of team profile data.

Table 5-16 – Group Profile Summary

| Participant | Q1 | Q2 | Q3 | User Score | Team Score (Max 15) |
|-------------|----|----|----|------------|---------------------|
| T2          | 4  | 5  | 3  | 12         | 12                  |
| T5          | 4  | 4  | 4  | 12         |                     |
| T3          | 4  | 4  | 3  | 11         | 10                  |
| T6          | 5  | 1  | 3  | 9          |                     |
| T1          | 3  | 3  | 2  | 7          | 8.5                 |
| T4          | 3  | 4  | 3  | 10         |                     |

Mean: 10

Standard Deviation: 1.8

It can be seen from the results that each team and individual were generally of intermediate skill.

### User Experience

The following 12 closed questions were given in the individual questionnaire:

1. Overall, the tool supported me to create process models which met the given business conditions
2. I was confident I had learned the underlying declarative modelling concepts
3. I was confident I had learned the underlying procedural modelling concepts
4. The time and effort required to learn the underlying declarative modelling concepts was reasonable
5. The time and effort required to learn the underlying procedural modelling concepts was reasonable
6. Working with the tool individually, I was confident in the accuracy of my interpretations of declarative models

7. Working with the tool individually, I was confident in the accuracy of my articulations of procedural models
8. The time and effort required to interpret declarative models was reasonable
9. The time and effort required to create/adapt/interpret procedural models was reasonable
10. The verification view was helpful during model development to help drive further development of the model
11. The information provided by the verification view was readily understandable and actionable
12. When interpreting the constraints I preferred to use the graphical model rather than the tabular display

The figure below gives a graphical representation of individual questionnaire results, showing the average (mean) response for each question. The complete breakdown for individual questionnaire results can be found in Appendix J.

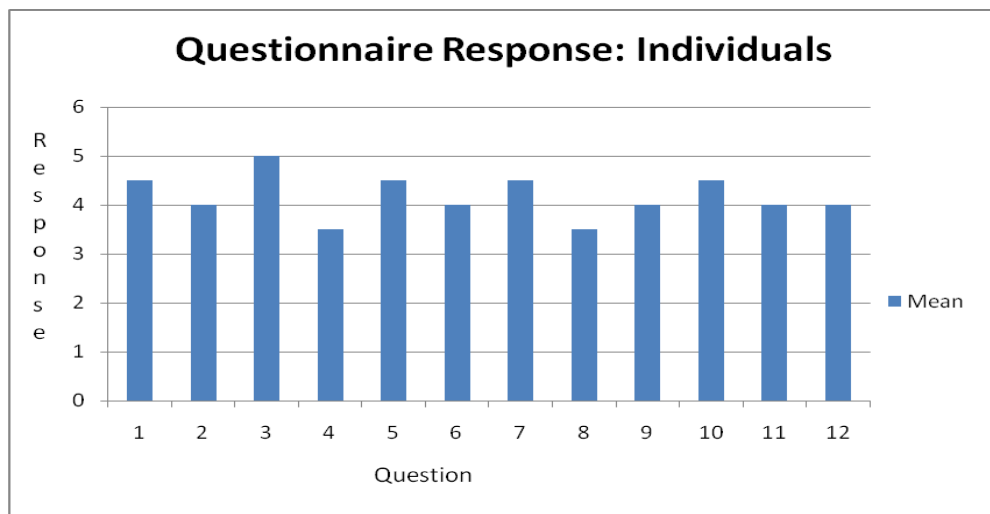


Figure 5-4 - Questionnaire results for Individuals

The following 15 closed questions were given in the group questionnaire:

1. Overall, the tool supported a team to jointly create models which met given business conditions (i.e. the system fostered reaching agreement and consensus)
2. I was confident I had learned the underlying declarative modelling concepts
3. I was confident I had learned the underlying procedural modelling concepts
4. The time and effort required to learn the underlying declarative modelling concepts was reasonable

5. The time and effort required to learn the underlying procedural modelling concepts was reasonable
6. Working with the tool in groups, when necessary, I was able to explain my understanding/point of view to others
7. Working with the tool in groups, when necessary, I was able to understand the points of view of others
8. Working with the tool in groups, I was confident in the accuracy of our interpretations of declarative models
9. Working with the tool in groups, I was confident in the accuracy of our articulations of procedural models
10. Working with the tool in groups, we were able to reach agreement throughout the build process
11. The time and effort required to perform interpretation of declarative models in groups (i.e. to get consensus of meaning) was reasonable
12. The time and effort required to perform create/adapt/interpret procedural models in groups (i.e. to get consensus of solution) was reasonable
13. The verification was helpful during model development to help drive further development of the model
14. The information provided by the verification view was readily understandable and actionable
15. When interpreting the constraints I preferred to use the graphical model rather than the tabular display

The figure below gives a graphical representation of group questionnaire results, showing the average (mean) response for each question. The complete breakdown for group questionnaire results can be found in Appendix J.

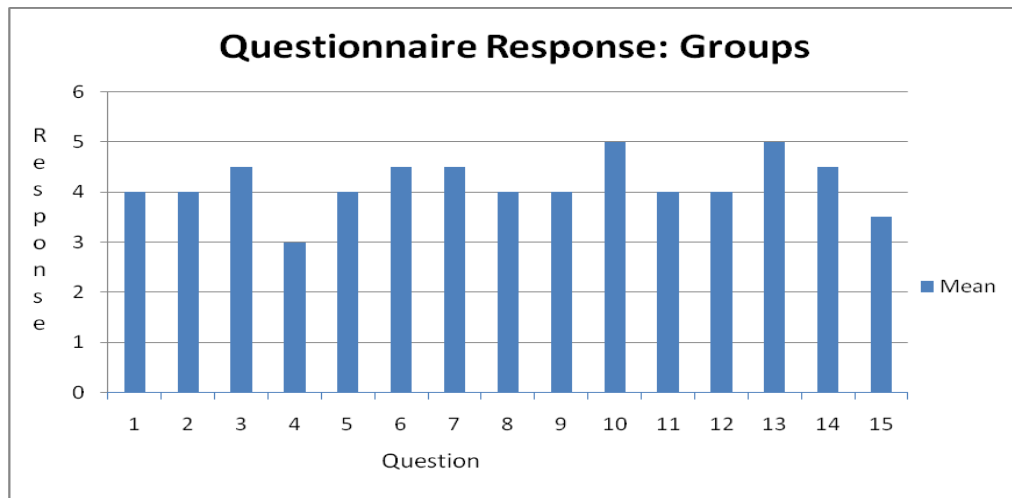


Figure 5-5 - Questionnaire Results for Groups

### 5.6.6 Other Feedback

The same theme emerged from the participants who provided further feedback. The visualisation of links becomes difficult on larger diagrams. This is the case for both procedural and declarative content. This is not a weakness of the underlying concept, rather the graphical engine.

A second theme emerged through feedback and during the training sessions. The design of the help documentation could be improved. It emerged that a larger, real world example would speed up the learning process.

## 5.7 Observations

This section will report on the observations made by the researcher throughout the training and the task exercises.

### 5.7.1 Problem Areas

A list of anticipated problematic areas was created. Instances of problems (learning and performance of approach concepts) in the training and task session were recorded. If problems identified were not anticipated, the nature of the problem was added to the list.

Instances of problems were identified through both the questions participants asked and through observation during the training and task sessions.

Table 5-17 – Researcher Observations During Task

| Area - events/problematic areas  | Training | Task |
|--|----------|------|
| <b>Anticipated</b>   |          |      |
| Constraint types   |          |      |
| Serial   | iii      |      |
| Fork   | i        |      |
| Include  | i        |      |
| Exclude  | i        |      |
| Precede  | iii      | iii  |
| Succeed  | iii      | i    |
| Strong and weak concept  | iii      |      |
| Type - instance concept  | iiii     | ii   |
| Multiple targets and target logic  | ii       |      |
| Procedural concepts  |          |      |
| Start and stop nodes   | iii      | i    |
| Task nodes   |          |      |
| Cycles of build, verify, adapt. what to do once built – directed to verify, process of correction – drill; down to individual constraints and correct  | iii      | ii   |
| Meaning of verification colours  | ii       | i    |
| Lengthy time to learn the different node action (drag, menus)  | iii      |      |
| <b>Emergent</b>  |          |      |
| Precede description with ‘upstream’ and succeed description with ‘downstream’ was confusing. User preferred with flow and against flow.  | i        |      |
| How get from one model to the other  | i        |      |
| Node instance names  | ii       |      |
| Equivalence of constraints (pro and dec) had to describe how different   | i        |      |
| Quality of the help/instructional materials came under scrutiny. A lot of jargon and not enough examples. Real world examples would aid understanding, need in addition to the conceptual description. | iii      |      |

It was observed that users mainly used the information available on verification view itself. The graphical model was only used when participants required a detailed view of the constraint (as information on the verification view is limited.)

### 5.7.2 Chat Transcript Analysis for Group Work

A chat facility forms part of the PowerMeeting system and is available to each plug-in. The transcripts generated by groups during the tasks were used as a window to view how group work progressed. In general the transcripts show that team members were able to discuss and refer to the model, articulate their position/opinion, to understand the position/opinion of others and to move forward with group consensus.

Transcript 1 showed how one group member was more confident with the system than the other. A promising outcome was how the system facilitated group work as one participant was able to help the other to understand the underlying concepts and how to use the tool.

What was shown in both transcripts 1 and 2 is how members of the group agreed on action to take and then took that action. Achieving group consensus is an important part of the framework described in Chapter 3.

The use of the given graphical declarative model was shown in transcripts 2 and 3. The model was used to build an entire procedural model in advance which was then checked using the verification view and corrected. This shows how participants were using the graphical model to build quality into the model rather than just relying on the verification tool to help generate a correct model.

Transcripts 1, 2 and 3 all demonstrate how groups operated in cycles; build - analyse (through chat and use verification tool) - agreement on next action. Again this is a key part of the framework discussed in Chapter 3. These results demonstrate the viability of tool support for its implementation.

## 5.8 Data Analysis

The absence of standard techniques for assessing the usability performance metric of a modelling language/tool means the usability study was a uniquely formulated usability study. The absence of benchmarks for comparison led to a pragmatic approach being taken to interpret the meaning of results.

This section will use the data collected from the usability study presented above to assess how far it supports the two hypotheses given in Section 5.5.2. Accordingly this section will be organised around those hypotheses, each hypothesis being supported by data from the task and questionnaire results.

**H1 - Working individually, users can successfully use a hybrid interactive approach to agile planning, as enabled by PowerMeeting and D2P.**



*Effective learning* - We propose that learning is a pre-requisite of performance. The effective performance measure will then used to assess effective learning for task data.

This criterion was targeted by questions 2 and 3 in both the individual and group questionnaire (group included as learning was performed individually). A mean response score of 4 and 4.75 out of 5 respectively, clearly demonstrates support.

*Effective performance* - Table 5-11 shows that on average participants could successfully complete 76% of the given modelling task. Although a predefined level of success was not defined, a pragmatic analysis approach would indicate support of the performance and learning criteria's.

This criterion was targeted by questions 6, 7, 10 and 11. Response scores of 4, 4.5, 4.5 and 4 respectively, clearly demonstrate support.

*Efficient learning* - Table 5-10 shows the average time spent learning before the user claimed to be comfortable to move forward was 59 minutes. Again no predefined time was targeted to denote success; instead a pragmatic approach is used. This average represents an encouraging result. Consider in industry that planned, taught training sessions can last for many hours and days.

This success factor was targeted by questions 4 and 5 in both the individual and group questionnaire (group included as learning was performed individually). A mean response of 3.25 and 4.13 out of 5 respectively, demonstrates support. It was the time and effort of learning the declarative modelling technique which was less strongly supported at 3.25.

*Efficient performance* - Table 5-13 shows the average amount of time to complete the task was 31 minutes. Using the predefined measure of success, we can say the average user completed the task in fast time.

This question was targeted by 8 and 9. A mean of 3.5 and 4 respectively, demonstrates support. It was the time and effort required to interpret declarative models which was less strongly supported at 3.5.

Overall, the data gained from this initial usability study supports H1. These results are show further promise given that participants were not extensively trained. Time to learn and interpret declarative models was the weakest result. As an overall impression of user experience with the tool, question 1 scored a mean response of 4.5.

**H2 - Working in groups, users can successfully use a hybrid interactive approach to agile planning, as enabled by PowerMeeting and D2P.**

*Effective performance* - Table 5-12 shows that on average participants could successfully complete 77% of the given modelling task. Again, although a predefined level of success was not defined, a pragmatic analysis approach would indicate support of the success criteria.

This question was targeted by 6, 7, 8, 9, 10 and 13 and 14. Mean responses of 4.5, 4.5, 4, 4, 5 and 5 respectively, clearly demonstrates support.

With the one distant training session, it was demonstrated how the tools are suitable to support training/learning in a collaborative fashion; chat was used to answer questions, pointer was used to guide the participant round the model etc.

*Efficient performance* - Table 5-14 shows the average amount of time to complete the task was 32 minutes. Using the predefined measure of success, we can say the average user completed the task in fast time.

This question was targeted by 11 and 14. Mean responses of 4 and 4.5 respectively, clearly demonstrates support.

Overall, the data gained from this initial usability study supports H2. As an overall impression of experience, question 1 scored a mean response of 4.0. This is a slight drop of over the experience of the individual but not a critically large difference. Caution should be taken with the extent of claims made, especially for group working. First of all, it is problematic to assume that the way formal constructs are used small groups will scale up to the way they are used in larger scale operations. Caution is required when results from small group studies are generalised to larger cooperative work arrangements. Furthermore,

the usability study focused on task work aspects of an interactive system, rather than investigate strongly the support provided for teamwork - the 'work of working together' (Baker, Greenberg et al. 2002).

## 5.9 Alternative Usability Study Designs

The evaluation could have targeted the approach on a conceptual level; taking the implementation out of the usability study. This would have taken the software prototype and group working aspect out of evaluation, both of which are important themes in this research.

The usability study task could have been designed differently e.g. ask user if given models were correct or not or provide a model and asked user to correct. This would have moved away from natural usage of the approach. This study tried to replicate natural usage in both interpretation and articulation.

Usability questionnaires (e.g. SUS (Tullis and Albert 2008)) could have been used. We wanted to investigate particular aspects. SUS is less suitable for an academic prototype which has been developed to answer specific research questions. Their strength lies in industrial strength systems nearing release.

## 5.10 Summary

This chapter has presented a summary of the evaluation component of a Design Science (Hevner, March et al. 2004) research project. Part of the improvement targeted by the approach presented in Chapter 3 and realised in Chapter 4 comes in balancing process flexibility and control. However, the real question is does that technique impact substantially upon usability?

The majority of the chapter has described the design, execution and results of a usability study addressing the usability of the D2P software prototypes. The initial findings were encouraging and suggested that semi-skilled users with a small amount of training are indeed capable of using a hybrid interactive modelling approach. This was shown to be the case when users worked as individuals and in small groups.

The results have contributed to flexible process support and beyond. However, it is instantly obvious that further work is required to extend the scope and content of the usability study towards design more realistically reflecting real world situations. To further move up the evaluation hierarchy, usage in real world environments themselves is a necessary step.

# Chapter 6

---

## Conclusions and Further Work

The end goal of BPM is to facilitate the effective coordination of all entities (tasks, people and information) involved in the execution of a business process. Process-orientation underpins the modern enterprise landscape and thus process-related research will always attract great interest due this dominance. Furthermore, the ‘wicked’ nature of providing flexible process support ensures its enduring challenge. When dealing with such a complex problem, progress is always gradual but systematic; the state of the art is advanced by step by step through the identification and exploration of process-related issues.

It is towards the goal of flexible process support that this thesis has made a contribution. The first challenge was to narrow the scope of this broad, encompassing problem. An opportunity was identified in the failure of current techniques and technologies to effectively provide IT support for the coordination of EKP. The main characteristics of such processes are human and knowledge-orientation, lack of long foresight when making plans, the fluidity of plans once made and the need to ensure process compliance. Furthermore, due to globalisation, the planning and performance of EKP is increasingly being performed by dispersed teams in distributed networked organisations. This exacerbates an already challenging problem.

By summarising the contributions of the study, Section 6.1 indicates how far this research study has addressed this problem. The most important limitations of the study are highlighted in Section 6.2. Finally, Section 6.3 closes the chapter with a comprehensive list of directions for further research.

### 6.1 Contributions of the Study

The approach sought to exceed prior research in three ways. Lets us consider the contribution made in terms of the original research questions:

*What critical issues and limiting factors can be identified in the current crop of PAIS in terms of their application to EKP?*

A problem space was generated which described EKP and the wider organisational forms which are likely to employ them. This problem space was used to appraise the fitness of a wide range of current tools and techniques which could potentially be applied to it. The appraisal extended to coverage of traditional groupware systems, traditional workflow systems, flexible workflow systems, interactive modelling systems and integrated systems. This led to the identification of several inhibiting issues. Systems were found to be lacking in one or more of the following areas:

- Inability to provide a practical, usable approach to provide a balance process control and flexibility.
- Limited or no collaboration opportunities.
- Fragmented tool sets for processes with varying support needs
- Cumbersome system deployment and system connection/disconnection paradigms.

The problems presented in Chapter 2 have arisen due to the mismatch between existing requirements and existing solutions. In defining a problem space the relevance of the research is clearly demonstrated as a current and open problem is addressed. Addressing each one of these challenges can generate business value and advance the state of the art in BPM. Comparison with previous studies shows that other work is currently being done in the same area and is addressing the problem in an ongoing research effort.

It is important to note, that in narrowing down the problem space and carving out a research question, the ‘wickedness’ of the problem has not been removed. How to use IS to provide better support for the planning and performance of EKP remains a truly complex problem as it straddles people, technology and business.

*What techniques and tools can help support users in the planning, performance and management of EKP with specific regard to any identified critical factors?*

The purpose of this research effort has been the development of technology and techniques that create new business value in the field of BPM by addressing current deficits in support. A novel approach based on graphical hybrid interactive models, flexible model management and collaboration was conceptualised. Hybrid graphical procedural and declarative models have never been used in this way before. The purpose of a graphical formulation of declarative models was to reduce the skill required for declarative model activation (articulation and interpretation) and thus represents an attempt to meet the human needs of flexible process support. The advantages of separation are that rules can be

modified independent from the process logic, thus leading to more readily flexible and adaptable business processes. In this way the applicability of the existing interactive modelling approach has been widened to cover environments where compliance to organisational rules is an important factor (such as with EKP). The effectiveness of the interactive modelling approach is thus further demonstrated.

The need and benefit of joint planning and analysis of work has been underestimated in BPM. Group working in the creation and subsequent identification and response to problems is fast becoming an important competence as multi-participant processes are common. This work helps realise that competence. A strategy for how different modes of collaborative working can form part of a PAIS was presented.

A key strength of this research project how the work presented tightly integrates with other research efforts. The framework demonstrates how the work presented tightly integrates with other research efforts as it is born out of existing work; through small adjustments to existing techniques and combinations of previously fragmented techniques a novel approach was formed and evaluated.

The way of working was realised in a collaborative, Web-based modelling system. D2P was delivered as a 'plug-in' for the PowerMeeting system. By leveraging the Web, the rapid connection and disconnection to a process support environment necessary to facilitate real-time organisations is achieved. It was shown how Web technology and standard web browsers can support full-fledged interactive real-time process-aware groupware with desktop like responsiveness and user experience.

It was the combination of hybrid interactive models, collaboration and Web technology which was ventured as a candidate approach to address the critical issues described above. While previous approaches neglected the integration of process support, collaboration and Web technology this approach aims on integration without sacrificing usability.

#### *Is the approach useful/usable?*

This thesis provides some much needed empirical assessment of the ability of semi-skilled users to perform modelling tasks using both procedural and declarative modelling techniques. It was shown through problem solving tasks performed in experiment conditions, that participants were able to interpret declarative models to generate

procedural models using D2P. Participants were able to learn the concepts of the approach and demonstrate that learning through performance.

Artefacts generated by a Design Science (Hevner, March et al. 2004) research approach can be validated through proof of concept. This was achieved with an implementation of the concept which was described in Chapter 4. This study then exceeds that requirement with the design and execution of a usability study.

The findings of this work have implications for related domains. For example, the underlying technique of using both graphical procedural and declarative models could be deployed for a workflow-oriented process support solution. This would serve to capture domain rules and also reduce the skill requirement of system administrators. The hybrid modelling approach could also be applied to the composition of Web services.

### **Research Communication**

A Design Science (Hevner, March et al. 2004) research approach relies on the comprehensive communication of research results. The primary communication of the research has been made through this thesis document. Other communication outlets which have formed an important part of the research process include conference and journal publications as well as presentations at various seminars.

## **6.2 Limitations and Scope for Further Work**

When assessing research it can be difficult to distinguish limitations from further work. Limitations exist within the scope of the research set by objectives. Further work exists outside this scope.

### **6.2.1 Limitations of the Study**

The limitations detailed below do form a barrier to take up of the approach as they highlight areas of immaturity. The limitations can be viewed through the lens of the Design Science (Hevner, March et al. 2004) research approach:



### **Problem Awareness**

Conceptual and technical business problems formed the primary driver and rationale of the artefacts which were developed. A basic understanding of the problem domain and use situation motivated development. Whilst this coverage goes beyond a superficial description, a thorough survey of the organisational, social and cognitive aspects of the work context would have improved problem formulation and contributed to a more focused system build based on more accurate representations. This could be achieved with diverse methods such as an analysis of user activities with discourse, documents, ethnography, cultural probes, user profiling techniques, focus groups, interviews, questionnaires and field studies.

### **Artefact Development**

Several factors were involved in the development of a solution addressing the business problem. Researcher intuition, skill and experience played an important role. Whilst, this is perfectly acceptable part of the Design Science (Hevner, March et al. 2004) methodology, it could have been improved through its augmentation with a user-centred design effort. Involving likely or typical end users in the development process would have added fresh perspective and design ideas.

### **Problem Relevance**

Firstly, the approach presented requires extra user effort to learn and use. The source of the required effort is likely to be the unfamiliar declarative modelling paradigm. Compounding this aspect is the current reliance on manual modelling which can be time-consuming and error-prone.

Secondly, the cost of graphical formulation of constraints is the lack of scalability. Declarative models can become complex and lose their visual appeal and understandability in more realistic use. Indeed, this problem is true of many others types of information which use graphical formulation. This effect can be reduced with advanced diagramming techniques.

### **Evaluation**

With a usability study various performance measures can be collected. However, the particular reasoning behind the measures is not. It must also be stressed that the study is undertaken in an artificial environment (taking participants out of a natural working

environment). This has an impact on measures taken and results may not necessarily transfer to a natural working environment.

### **Search Process**

Design Science (Hevner, March et al. 2004) research is ideally iterative on a large scale; evaluation results feed back into development in a continuous cycle. Due to resource constraints, this research consisted of a single round of large scale evaluation (discounting the continual researcher-led ‘mini-evaluations’). The findings from this evaluation instead of feeding into (re)development, formed part of the content for the further work section.

## **6.2.2 Scope for Further Work**

This section represents opportunities for further development of the research. The issues raised below do not impact on the research questions and contribution made.

### **Areas for the development of the approach and prototype**

The project concentrates on the technical design and implementation of a flexible process support tool. Scope for further work includes but is not limited to:

- Improvement in the validation of declarative models. Declarative models can be created with internal conflicts. Currently, knowledge workers are left with the burden of ensuring the correctness and removing conflicts in the models through manual inspection. Work should be carried out to address this issue, first with a conceptual technique and then its realisation through additional automated computational supports addressing issues such as correctness and redundancy.
- The current prototype uses atomic models only. Prototype development should implement this design and improve on it with ways to create, compose, manage and navigate hierarchical process structures.
- Low level model management facilities such as undo, cut, copy, paste, version control, roll-back, change traceability, access control, improved visualisation (graph layout algorithms) and zooming capabilities can improve the effectiveness of model interaction. In addition, the usage of change patterns can help reduce error prone and complex manual interactions which the approach currently rely on.
- High level model management facilities such as template creation (with meta-data such as name, author, category and user comments), template repository, repository searching and access control.

- Improving decision support. Currently participants involved in the configuration of process models need to possess expertise both in the application domain and in the modelling language employed. Techniques such as wizards, Artificial Intelligence assisted suggestions and reuse of past solutions could be used reduce this modelling burden.
- Tighter integration with external tools and Web services for automated execution of tasks can improve and widen applicability. One direction is the provision of alternative communication channels such as video and voice (e.g. with closer integration with Skype.) A second direction is integration to Web service discovery and consumption services. The hybrid modelling approach can then apply to the coordination of Web services.
- Addressing the dynamic change problem. If a template is changed, how should that change impact 'live' instances? Techniques should be put in place to deal with template change.
- Interactive meta-models. A mechanism which allows the simple creation and adaptation of meta-models is required to rapidly tailor the approach for organisational and project needs. Hard-coded meta-models are not conveniently modified and require the attention of professional developer. The declarative language can be developed in many areas:
  - Assert the criticality of constraints e.g. preferred, mandatory.
  - Relative temporal rules e.g. task A starts two days after B ends.
  - Dynamic rules which use contextual data such as data associated with the process e.g. if task A is two days overdue, task B must follow task A.
  - Richer resource rules could be developed with an organisational model e.g. task C must be performed by Johns supervisor or Task C must be performed by someone from a different team than Jake. EKP are highly dependent on the skills, knowledge and experience of the people carrying them out. Flexible and compliant assignment of appropriate knowledge-process participants is crucial to accomplishing effective knowledge processes. Therefore resource management is a growing concern in BPM. The resource perspective centres on the modelling of resources and their link with procedural information. Overwhelming emphasis has been placed on the control-flow in this and other contemporary modelling notations such as BPMN.

- Need for integrated Project Management support. The current procedural model cannot articulate deadlines or the assign the effort required for tasks for example.
- Application platform development. Distribution of application and process data to multiple servers and further examination of speed, reliability and assess the impact of load.

### **Directions for Improved Evaluation**

Experimentation can be further used to assess:

- The impact of grammar realisation. The same underlying grammar can be achieved in different ways. The experiment could address and improve grammar aspects by adjusting the shallow realisation of the grammar (e.g. text v graphical) and examining impact on result.
- The impact of script properties. For example, model complexity (at what point does understandability drop off?) and visual layout strategies (what is their impact on understandability?)
- The ease of use and learning of system. Further work is necessary to gather opinion on if the interface is clear and understandable, simple and intuitive. Gathering opinion on the comfort, pleasure, format of output, the layout design and display of the output contents, volume of output and the amount of the information given to a user can contribute to improving usability of future prototypes.
- The impact of user and group characteristics on activation of the approach. These characteristics can be group size, skill level, psychological characteristics and experience level for example.
- The impact of different training methods (e.g. remote versus local.)

Each kind of evaluation has its strengths. The positivist studies with predefined hypotheses described in Chapter 5 miss the richness of reality. In usage, the tool and users form extremely complex socio-technical information system. Examining the interactive modelling approach with longer-term, interpretive fieldwork in real organisations will provide a fresh lens on its effectiveness. Fieldwork can provide empirical insight complex and incompletely understood realities of work practice which is missing from this thesis. Each of the above opportunities for further evaluative study could be investigated with interpretive field work. In addition, the usefulness measure of the approach can particularly benefit from longer term field work:

- As the software prototype was geared to address specific research questions, the actual usefulness of the approach was problematic to evaluate. Evaluating perceived usefulness from the user perspective was also problematic. Usefulness addresses issues such as does the approach allow the user to work more quickly, improve job performance, increase productivity and makes job easier. Essentially, usefulness from the user's perspective asks the user's judgment about the relevant balance between the cost and the considered usefulness. Gaining such data would require prototype development into a larger real world tool and then its evaluation in real world case studies. Furthermore, contemporary design principles emphasise measuring user reactions throughout the entire design process which was lacking.

# Appendix A

---

## Justification of Declarative Expression Technique Choice

Generally constraints establish rules on the composition (the values that a variable, or a combination of variables may take) of a target artefact. This pattern can be found in countless domains. Table A.1 gives but a few examples.

Table A.1 – Simple Constraint Based Problems

| Target Artefact             | Constraint   |
|-----------------------------|--|
| A natural number            | x is greater than 2<br>x is less than 6                |
| A travel route              | Destination is Manchester<br>Use the M60 is prohibited |
| University course selection | Any 3 out of the given 5 courses can be selected       |

We apply this mechanism to process support; the procedural model is the target artefact, the declarative model establishes constraints (made up of at least one constraint (Boolean) expression).

A constraint expression technique can be evaluated by the extent to which it affords the three characteristics defined below:

- *Expressiveness* - representational power of a language (i.e. the ability to represent domain phenomena).
- *Technical Actor* - the scope for computational support in interpretation and articulation of the language.
- *Social Actor* - the scope for human interpretation and articulation of the language.

These are challenging characteristics for a language to achieve and are not often found in close companionship. Constraints can be expressed with different techniques. We shall use the characteristics to appraise potential constraint expression techniques. Two important classifications of technique are:

- *Explicitness* - the extent to which the language prescribes specific resolutions and what user action is required to get that resolution;
  - *Extensional* - explicitly enumerating all of the allowable states or values that may be held by an artefact. The role of the user is to choose amongst predefined solutions.

- *Intentional* - implicitly restricting the artefact to those states covered under some logical expression of what constitutes a valid interpretation of the domain semantics. The role of the user is to generate there own solution.
- *Specialisation* - the extent to which the language is geared to a specific domain/scope of applicability;
  - *General* - widely applicable e.g. natural language/free text. These languages are highly expressive.
  - *Specific* - domain-oriented e.g. Web service composition or process modelling. These languages are less expressive.

Controlled languages such as OCL and first order logic would occupy the middle ground in terms of specialisation.

These characteristics can be combined to create a matrix with zones of expression type as pictured in Figure 3-6.

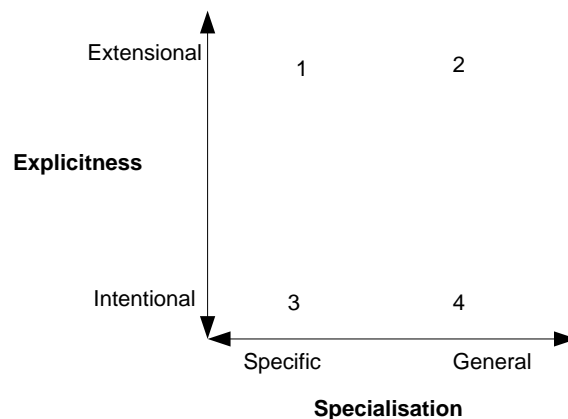


Figure A-1 - Dimensions of Constraint Specification

We shall briefly examine each area with respect to the characteristics given above to provide the rationale behind the choice of declarative modelling language.

#### Areas 1, 2 and 4

Zones 1, 2 can be discounted. EKP have a large, often infinite, number of valid solutions. This means it is impractical often impossible to enumerate each one in advance. Area 4 is as readily looked over too. The further right on the specialisation spectrum the greater the expressiveness of the language. For languages such as natural language, the scope for imprecision in expression and ambiguity in social actor interpretation, is heightened. In addition, limited computational supports are afforded. Controlled languages are looked

over as although expressive, can be difficult for social actors without a strong mathematical background. As this characteristic cannot be guaranteed in end users of PAIS, it is disregarded.

### **Area 3**

A specific language affords precise specification, strong scope for computational support and scope for ease of social actor articulation and interpretation.

It has been shown that an abundance of low level primitives acts as a barrier to programming (Lewis and Olson 1989). That is to say, it is generally accepted that the higher the expressive power of the language, the greater the burden is of learning that language for the social actor.

To address this problem, it has been suggested that development environments use ‘task-specific’ languages (Nardi and Johnson 1994). A smaller, targeted language can maintain a closer alignment between the problem (real-world phenomena) and solution (modelling language) domains. In other words, the strategy relies on a less expressive language with the advantage of a reduced burden of learning and use due to less construct excess. We advocate a domain-oriented approach to constraint specification. This language could be sector specific e.g. healthcare, banking, insurance etc.

To complement a domain-oriented approach, we advocate pattern usage. Patterns are reusable solutions in some domain. The nature of the solution is abstract and can be applied repeatedly (Alexander 1979). Each coherent set of patterns found in a domain can form a language. The declarative language presented is then a small set of patterns useful in process-oriented domains - a test-bed language is employed to examine the concept of hybrid models introduced in the previous section. Although the language is small, considerable scope for flexibility in application is still maintained. The intentional, specific, predefined language implemented will afford articulation of static task and resource aspect constraints.

Using a specific language, expressivity is sacrificed; there are always constraints that cannot be expressed when using predefined language. However, it is important to note, that it is not the set of constraints which are important, it is the concept of interactive hybrid models.



# Appendix B

---

## Pseudo Code: Verification Algorithms

### Serial/Fork (X)

```
Check = All instances of (task type X) in P
If Check = Empty then Return Black
Else
    For i = 1 to count (Check)
        Str = Upstream (Check (i))
        Str = Str + DownStream (Check (i))
        Out = P - Str
        For j = 1 to count (Out/Str)
            If X.contains (Out/Str (j).task type)
                Return pink
    Return green
```

### Exclude (X, Y)

```
Targets = set of task instances in P with task type (tasktype)
Check = All instances of (task type X) in P
If Check = Empty then Return Black
Else
    St = pink
    If mode = each
        For k = 1 to count (targets)
            For l = 1 to count (P)
                If P(l).type = targets(k).type then return pink

        St =amber

    Else if mode = any
        For k = 1 to count (targets)
            Count = 0
            Found = false
            For l = 1 to count (P)
                If P(l).type = targets(k).type then found = true
            If found = true then count = count + 1
        If count = < count(targets) then st = amber

    Else
        For k = 1 to count (targets)
            Count = 0
            Found = false
            For l = 1 to count (P)
                If P(l).type = targets(k).type then found = true
            If found = true then count = count + 1
        If count != 1 then st = pink
        Else st = amber

    Return st
```

## Include

Members = set of members

Targets = set of targets

If mode = weak

St = pink

For i = 1 to count (members)

For j = 1 to count (P)

If P(j).type = members(i).type

If model = any

For k = 1 to count (targets)

For l = 1 to count (P)

If P(l).type = targets(k).type then st = amber

If model = each

Found = false

For k = 1 to count (targets)

For l = 1 to count (P)

If P(l).type = targets(k).type then found = true

If found = false then return pink

St =amber

If model = one

st = amber

Co = 0

For k = 1 to count (targets)

For l = 1 to count (P)

If P(l).type = targets(k).type then Co = Co + 1

If Co = 0 return pink

Else if Co > 1 return pink

Return st

If mode = strong

St = pink

For i = 1 to count (members)

For j = 1 to count (P)

If P(j).type = members(i).type

If model = any

For k = 1 to count (targets)

For l = 1 to count (P)

If P(l).type = targets(k).type

st = amber

Up.Remove (l)

If model = each

Found = false

For k = 1 to count (targets)

For l = 1 to count (P)

If P(l).type = targets(k).type

found = true

P.Remove (l)

If found = false then return pink

St =amber

If model = one

st = amber

Co = 0

For k = 1 to count (targets)

For l = 1 to count (P)

If P(l).type = targets(k).type

Co = Co + 1

P.Remove (l)

If Co = 0 return pink

Else if Co > 1 return pink

Return st

### **Precede/Succeed (X, Y)**

Members = set of members

Targets = set of targets

getStream = Upstream (for precede) Downstream (for succeed)

If Check = Empty then Return Black

If mode = weak

St = pink

For i = 1 to count (members)

For j = 1 to count (P)

If P(j).type = members(i).type

stream = getstream (P(i))

If model = any

For k = 1 to count (targets)

For l = 1 to count (Up)

If Up(l).type = targets(k).type then st =  
amber

If model = each

Found = false

For k = 1 to count (targets)

For l = 1 to count (Up)

If Up(l).type = targets(k).type then found = true

If found = false then return pink

St =amber

If model = one

st = amber

Co = 0

For k = 1 to count (targets)

For l = 1 to count (Up)

If Up(l).type = targets(k).type then Co = Co  
+ 1

If Co = 0 return pink

Else if Co > 1 return pink

If mode = strong

St = pink

For i = 1 to count (members)

For j = 1 to count (P)

If P(j).type = members(i).type

stream = getstream (P(i))

If model = any

For k = 1 to count (targets)

For l = 1 to count (Up)

If Up(l).type = targets(k).type  
st = amber  
Up.Remove (l)

If model = each

Found = false

For k = 1 to count (targets)

For l = 1 to count (Up)

If Up(l).type = targets(k).type  
found = true  
Up.Remove (l)

If found = false then return pink

St =amber

If model = one

st = amber

Co = 0

For k = 1 to count (targets)

For l = 1 to count (Up)

If Up(l).type = targets(k).type  
Co = Co + 1  
Up.Remove (l)

If Co = 0 return pink

Else if Co > 1 return pink

Return st

# Appendix C

---

## Placing D2P Amongst Other Groupware Applications

With the integration of a group working element, the approach to flexible process support takes the form of a process-oriented groupware application. To understand its place amongst other group working applications, we can position it on several spectrums:

### **Application (business - fun)**

This spectrum represents what the application is primarily used for; a professional business purpose, a social application or purely for entertainment. The approach is a professional business application.

### **Participants (unknown - known)**

This spectrum represents the degree of familiarity between participants. This could range from completely anonymous through to very close acquaintances such as family and friends. It is likely that in the approach, participants will be members of a virtual team or co-located. Relatively closed ensembles of people will perform the work, all working towards common business goals.

### **Participation (wide - narrow)**

This spectrum represents the size of typical participation. Participation means taking part in joint activities for the purpose of reaching a common goal. The size of participation then refers to the number of people who work together on some common goal. Participation may be wide or narrow. It can be suggested that participation will differ depending on the artefact being worked on. Participation at the system and meta-model development level is likely to be wide to capture various requirements from users of system. Participation at the type model level is likely to be narrower, but still involving various stakeholders to develop a reusable type model. At the instance model level, participation could be narrower again as small groups of people collaborate on particular pieces of work. It has been shown then that although the participation requirement is narrower in scope than say for example large scale knowledge creation ventures such as wiki, it is still wide due to involving people up and down the company in terms of hierarchy and across the company in terms of geographical location.

# Appendix D

---

## Procedural Content: Editing Rules

### **Removing content**

P has the notion of a front line. Components of the P which are either on (ongoing work) or behind (completed work) the front line cannot be changed. Components in front of the line can be changed as this is only planned work.

Delete task node:

- On - not allowed (ongoing)
- Behind - not allowed (cancelled or completed)
- In-front - allowed (waiting , ready, cancelled)

Delete task link:

- Behind (Fired) - not allowed.
- In-front (Not fired) - allowed

Delete actor node/link

- On - not allowed (ongoing)
- Behind - not allowed (cancelled or completed)
- In-front - allowed (waiting , ready, cancelled)

# Appendix E

---

## The Technology behind PowerMeeting

The early Web was based on traditional client-server architecture. A Web server runs constantly waiting for HTML page requests from Web clients made via HTTP. Once a request is made, the client waits for a response in order to refresh its entire display (Web browser). It is problematic to base Web-based synchronous groupware on this architecture for the following two reasons:

1. *Coarse grain data operations* – client requests are answered with files or pages which are used to update the entire display of the Web browser. A practical Web-based synchronous groupware system would require update/refresh cycles operating at a finer grained data object level.
2. *Unidirectional communication* – a basic requirement for synchronous groupware is two-way communications between Web client and Web server (to send data changes made at the client and receive data changes made at other clients). Communication is one way as there is no built in support for a server to notify a client.

### **Addressing the grain of data operations**

AJAX (Garrett 2005) can be used to realise fine grain level data management (a Web browser can fetch pieces of data which are required to update only part of the user interface rather than the whole page). With AJAX, an XMLHttpRequest JavaScript object can make asynchronous HTTP requests to the server, when the data returns they will be used to update only the affected HTML elements on the DOM Tree. When used with CSS, such elements can be graphical widgets and content views. Thus AJAX makes a big contribution to creating a rich user experience.

### **Addressing Bidirectional communication**

Two way communications becomes possible using AJAX Push. The traditional polling technique for two way communication uses repeated client requests to check for updates from the server. This is inefficient as most requests will return negative. Also it is possible that updates can be slow as refreshes have to wait for next request. AJAX Push improves on this by employing a long polling technique. The client makes a request to the server. The request is kept open until new data is available from the server to update the browser.

Once dispatched to the client, the request connection is terminated and a new request is immediately opened to repeat the cycle. This technique requires a HTTP connection for each collaborating client. This connection is a thread on the sever side. Frequently there is a limit to the number of threads which can be opened on the sever side. To combat this, a technique known as continuation is employed. When a given number of connections is reached, the server will freeze an existing connection that is waiting for a response and return it to the thread pool, when data arrives the thread will be woken up to resume communication.

### **Emergent Challenge: Addressing Rapid Development**

The lower level technological challenges for Web-based synchronous groupware have been met. However, the techniques and frameworks required to rapidly develop Web-based synchronous groupware have been missing. Rapid development of Web-based groupware applications is difficult due to the multiple technologies involved; JavaScript, HTML, DOM, CSS etc. Other difficulties include the handling of browser incompatibilities, complex concurrent programming techniques, database integration (if persistent data is required), data replication and consistency across clients, servers and databases.

A governing principle of PowerMeeting was that it should facilitate quick and simple development of Web-based groupware applications. PowerMeeting employs an AJAX driven approach to enable synchronous Web-based groupware. In addition PowerMeeting implements framework which enables the rapid development of synchronous Web-based groupware by semi-skilled developers. PowerMeeting realises this principle by combining further technologies to build a readily extended, open system.

- *Google Web Toolkit - GWT* (Johnson 2006) allows developers to write their front end applications in the familiar Java language using graphical interface libraries similar to JavaSwing. The GWT compiler converts java classes into browser compliant HTML and JavaScript (a single thread in a Web browser). GWT has a rich set of GUI widgets and helps realise applications with features and functionality more associated with traditional desktop applications.
- *Commonground* - Commonground provides the underlying services such as session management, user management, replication and persistency management necessary for synchronous groupware applications. The server side keeps and event list for each online client. Whenever events are placed into the list by the server, they are retrieved immediately by the client. CRUD operations are classified into read-only

and update operations. Read-only operations are handled transaction generated by a client are performed locally. Update transactions generated by a client are sent to the server. At the server, if no conflicts are detected (a shared Set structure with a very low level of conflict rate in parallel adding and removing of its elements), to ensure persistency, the change is recorded in the database. The change is then broadcast to the clients in the same session. This is accomplished by placing the update in the event list for each client, which is immediately fetched by the client. The change notification is realised at object attribute level to minimise the potential conflicts and provide a fast user-to-user response time for their cooperative interaction.

PowerMeeting unites the technologies described into a coherent, usable framework. Hotspots are left open into which applications can be plugged into. Applications can be rapidly developed and plugged into to take advantage of the PowerMeeting group working facilities and Web presence. As underlying services are provided, Java developers can focus on task structure, graphical user interface design and computational support on task structure. In this way, the PowerMeeting environment can be used to help realise the Service Web.



# Appendix F

---

## Information pack for Participants of Usability Study (Individual Version)

### Usability Study: Overview and Instructions

The study addresses the usability of the D2P process support tool. The study should last around one hour and is divided into two sessions; a training session and a task session. All participants should undertake the training session. At the end of the task session you will be asked to complete a questionnaire regarding your experience.

### Training Session

This is a relaxed session where participants will be introduced to the usability study and learn how to use the tool. At the end of the session, participants should have a good idea of how the tool works. A practise task, similar to that which will be used in the study, will be provided. Participants are free to ask any questions throughout the session.

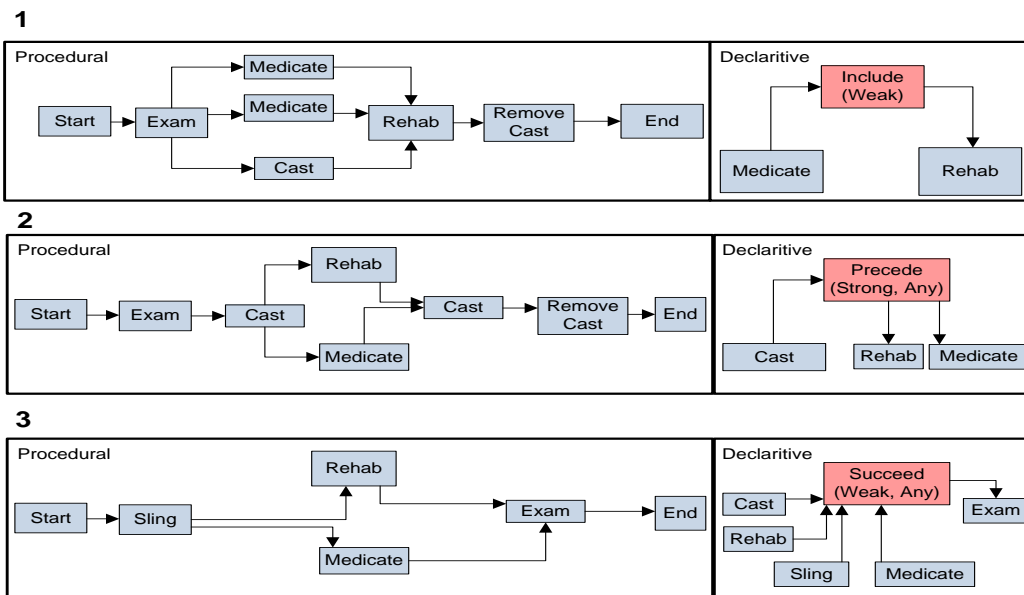
### Instructions for training:

1. Log onto the system using your first initial followed by your surname for the username and your allocated room name.
2. Use the create menu at the top left of PowerMeeting to create a copy of the template named 'D2P Training'. Open the model by selecting it from the work list.
3. On the D2P tool, navigate to the view menu. First read through Help 1 (gives an overview of the underlying goals and techniques of the approach) and then Help 2 (gives an overview of the tool which implements the above techniques.)

The training task is based around a simple treatment scenario involving the following task types:

| Task Type   | Description   |
|-------------|---|
| Examine     | The patient is examined by a doctor                       |
| Cast        | A cast is applied to the patients arm.                    |
| Remove Cast | A cast is removed from the patients arm.                  |
| Sling       | A sling is applied to the patients arm.                   |
| Medicate    | Medication is prescribed to the patients b the doctor     |
| Rehab       | Patients in a rehabilitation session with physiotherapist |

4. Below are three constraint models, each with a procedural resolution. Test your knowledge by reasoning if the procedural model is valid or invalid. The answers and explanation appear at the end of the document.



- Once more familiar with the way of working, create a procedural model of your own design, which satisfies the constraints in the template. Use the verification tool to view the correctness of the model you have created.

### Task Session

The task uses a ‘toy scenario’ to give a general idea of how the approach may be used. For the task, a declarative (constraint) model and case data is provided. From this material, you should generate a valid procedural model (i.e. satisfies the constraint model and given case data).

Read through the given task information and follow the instructions. You are free to use all help documents and ask any questions.

### Task: Broken Leg Rehabilitation

After the immediate emergency action (surgery and cast) has been performed, the usual care time for a broken Tibia or Fibula is 12 to 16 weeks. An orthopaedic doctor will make an assessment. This bone specialist will plan further follow-up appointments and rehabilitation as necessary. Very frequently, rehabilitation will involve a Chartered Physiotherapist to restore the range of ankle and knee movement, and to restore the muscle strength that is lost during the immobilisation period.

Our very simplified core process can be made up of the following task types. The provided template will arrange these task types to form a constraint model. The template will be provided in graphical form in the PowerMeeting system and also in tabular form (see after the instructions).

| Task Type | Description   |
|-----------|---|
| Assess    | Post surgery, an examination is performed to assess the needs of rehabilitation. Assessments can be performed by both orthopaedic doctors and physiotherapists              |
| Interim   | Interim examination's form a progress check various decisions are made here e.g. whether to remove the cast, weight bearing checks or whether to locate the patient at home |
| Plan exit | Hospital staff arrange how the patient will be transferred home – this could be a service provided the hospital if the case warranted it.                                   |

|             |  |
|-------------|--|
| Practical   | A medical professional will take the patient through practical day to issues such as diet, how to relieve swelling and infections, appropriate clothing and sanitation guidance. |
| Physio      | The physiotherapist performs various exercises with the patient; hydrotherapy exercises, manual therapy. In addition, the mental attitude of patient is addressed.               |
| Exit        | The patient is transferred to nominated accommodation  |
| Remove Cast | The cast is removed from the patient   |

### Case Data

1. The doctor and physiotherapist each perform an assessment
2. Two physio sessions are scheduled after the physios assessment
3. An interim examination is scheduled after the physio sessions.
4. The result of the interim examination is to remove the cast. After remove, the patient is located at the nominated place

### Instructions for Task:

1. Log onto the system using your first initial followed by your surname for the username. A room name will be provided.
2. Make a copy of the template named 'D2P: Rehabilitation'. Open the model by selecting it from the work list. The template is ready in the declarative modelling pane.
3. Build a procedural model (in the procedural modelling pane) which satisfies the given case data and the constraints given in the declarative model.
  - a. *You will need to consider both case data and constraint model to build a valid procedural model.*
4. Once happy with the process model, verify the model to check its quality. If errors are found, work on the process model to try and resolve the errors.
5. Once completed, inform the researcher.

| Constraint Type | Name | Mode              | Triggers                              | Targets              |
|-----------------|------|-------------------|---------------------------------------|----------------------|
| Serial          | Ser1 | -                 | -                                     | physio               |
| Fork            | For1 | -                 | -                                     | assess               |
| Precede         | Pre1 | Weak, each        | assess                                | practical, interim   |
| Precede         | Pre2 | Weak, each, chain | plan exit                             | exit                 |
| Precede         | Pre3 | Strong, each      | exit                                  | practical            |
| Succeed         | Suc1 | Weak, each        | plan exit, practical, interim, physio | assess               |
| Succeed         | Suc2 | Weak, each        | remove cast                           | interim              |
| Succeed         | Suc3 | Weak, each        | exit                                  | Plan exit, practical |

### Answers:

- 1/Valid – two medicate task instances are present. A rehab task instance is also present. As the mode is weak, one rehab task instance satisfies the constraint. If the mode was strong, another rehab task instance would be needed to make the model valid.
- 2/Invalid – In the upstream of each cast task instance, should be any of rehab and medicate. The first task instance has rehab and medicate in its upstream. The second cast task instance does not, so the model is invalid.
- 3/Invalid – in the downstream of any task instance of sling, cast, medicate or rehab should be an exam task instance. The instances of sling, rehab and medicate do not have an exam in the downstream, so the model is invalid.

# Appendix G

## Questionnaire (Group Version)

### Part 1 - Personnel Data and Background

Name:.....

Student Number (if applicable):.....

1/Rate your theoretical knowledge of business process modelling in any context (e.g. any techniques such as UML for IS development, BPML for process modelling or any process graphs (e.g. Petri nets) for process support)

Weak

Strong

|  |  |  |  |  |
|--|--|--|--|--|
|  |  |  |  |  |
|--|--|--|--|--|

2/Rate your practical experience of business process modelling in any context (e.g. where apply theoretical knowledge to perform modelling tasks)

Weak

Strong

|  |  |  |  |  |
|--|--|--|--|--|
|  |  |  |  |  |
|--|--|--|--|--|

3/Rate your practical experience of process aware tools in any context (e.g. workflow, groupware, project management tool)

Weak

Strong

|  |  |  |  |  |
|--|--|--|--|--|
|  |  |  |  |  |
|--|--|--|--|--|

### Part 2 - Experience and Perceptions (Please mark one box only):

1/Overall, the tool supported a team to jointly create models which met given business conditions (i.e. the system fostered reaching agreement and consensus)

Strongly Disagree

Strongly Agree

|  |  |  |  |  |
|--|--|--|--|--|
|  |  |  |  |  |
|--|--|--|--|--|

2/ I was confident I had learned the underlying declarative modelling concepts

Strongly Disagree

Strongly Agree

|  |  |  |  |  |
|--|--|--|--|--|
|  |  |  |  |  |
|--|--|--|--|--|

3/ I was confident I had learned the underlying procedural modelling concepts

Strongly Disagree

Strongly Agree

|  |  |  |  |  |
|--|--|--|--|--|
|  |  |  |  |  |
|--|--|--|--|--|

4/The time and effort required to learn the underlying declarative modelling concepts was reasonable

Strongly Disagree

Strongly Agree

|  |  |  |  |  |
|--|--|--|--|--|
|  |  |  |  |  |
|--|--|--|--|--|

5/The time and effort required to learn the underlying procedural modelling concepts was reasonable

Strongly Disagree

Strongly Agree

|  |  |  |  |  |
|--|--|--|--|--|
|  |  |  |  |  |
|--|--|--|--|--|

6/Working with the tool in groups, when necessary, I was able to explain my understanding/point of view to others

Strongly Disagree

Strongly Agree

|  |  |  |  |  |
|--|--|--|--|--|
|  |  |  |  |  |
|--|--|--|--|--|

7/Working with the tool in groups, when necessary, I was able to understand the points of view of others

Strongly Disagree

Strongly Agree

|  |  |  |  |  |
|--|--|--|--|--|
|  |  |  |  |  |
|--|--|--|--|--|

8/Working with the tool in groups, I was confident in the accuracy of our interpretations of declarative models

Strongly Disagree

Strongly Agree

|  |  |  |  |  |
|--|--|--|--|--|
|  |  |  |  |  |
|--|--|--|--|--|

9/Working with the tool in groups, I was confident in the accuracy of our articulations of procedural models

Strongly Disagree

Strongly Agree

|  |  |  |  |  |
|--|--|--|--|--|
|  |  |  |  |  |
|--|--|--|--|--|

10/Working with the tool in groups, we were able to reach agreement throughout the build process

Strongly Disagree

Strongly Agree

|  |  |  |  |  |
|--|--|--|--|--|
|  |  |  |  |  |
|--|--|--|--|--|

11/The time and effort required to perform interpretation of declarative models in groups (i.e. to get consensus of meaning) was reasonable

Strongly Disagree

Strongly Agree

|  |  |  |  |  |
|--|--|--|--|--|
|  |  |  |  |  |
|--|--|--|--|--|

12/The time and effort required to perform create/adapt/interpret procedural models in groups (i.e. to get consensus of solution) was reasonable

Strongly Disagree

Strongly Agree

|  |  |  |  |  |
|--|--|--|--|--|
|  |  |  |  |  |
|--|--|--|--|--|

13/The verification was helpful during model development to help drive further development of the model

Strongly Disagree

Strongly Agree

|  |  |  |  |  |
|--|--|--|--|--|
|  |  |  |  |  |
|--|--|--|--|--|

14/The information provided by the verification view was readily understandable and actionable

Strongly Disagree

Strongly Agree

|  |  |  |  |  |
|--|--|--|--|--|
|  |  |  |  |  |
|--|--|--|--|--|

15/When interpreting the constraints I preferred to use the graphical model rather than the tabular display

Strongly Disagree

Strongly Agree

|  |  |  |  |  |
|--|--|--|--|--|
|  |  |  |  |  |
|--|--|--|--|--|

**In the space below feel free to write any comments regarding the D2P concept and tool:**

# Appendix H

---

## D2P Help Documents

### D2P Help - Part 1

This document is intended to give an overview of the D2P approach for process support. For greater insight into the tool which implements the approach, refer to Help - Part 2.

#### Contents

#### [Goal - Flexible Process Support](#)

#### [A Model-Driven Approach](#)

#### [Procedural Models](#)

#### [Declarative Models](#)

- [Task Types](#)
- [Serial](#)
- [Fork](#)
- [Include](#)
- [Exclude](#)
- [Precedes](#)
- [Succeeds](#)

### Goal - Flexible Process Support

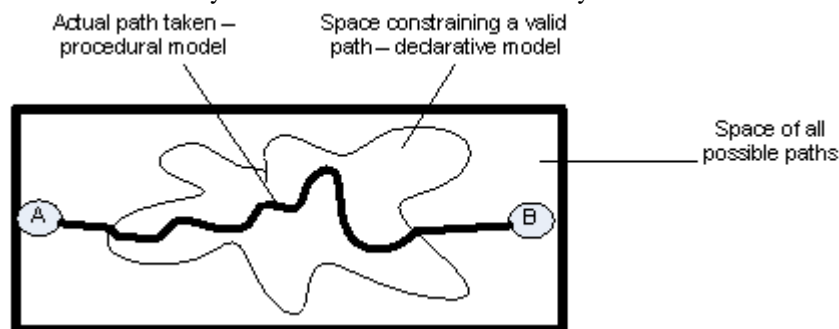
The primary goal of the work is to help people plan and perform their processes in a way which balances flexibility (allowing users the freedom to plan their own work) and control (ensuring that work complies with organisational regulations). The work supported often recurs e.g. a company's purchase process (different purchases will be made, each following a similar process).

### A Model-Driven Approach

Graphical models are used to help users visually plan, perform, analyse and adapt their work. With D2P, a process model can be made up of two types of content, each having a specific purpose:

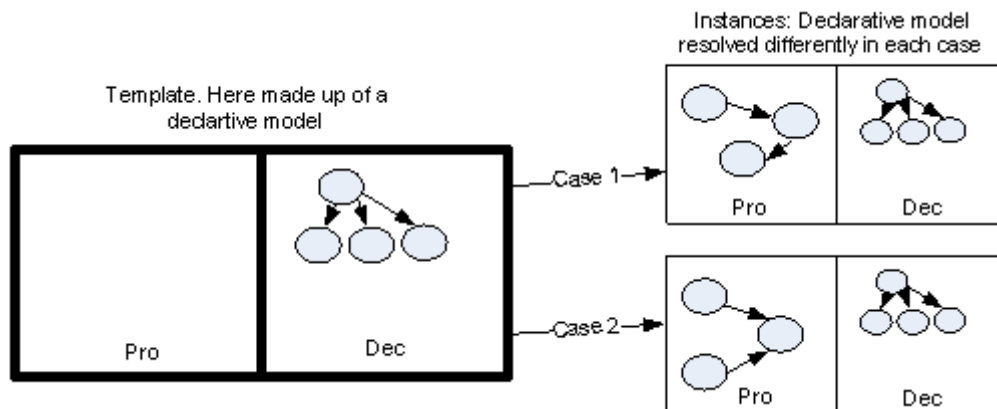
- A procedural model represents the actual work that will be performed. Usually created, analysed and executed by the work performer.
- A declarative model asserts properties which should hold in the procedural model (e.g. restrictions on the arrangement of tasks). Usually created prior by a compliance specialist.

Speaking conceptually, imagine the entire process model as a map and the goal is to get from A to B. The declarative model captures any restrictions on the possible routes. The procedural model captures the actual route we will take. In the below figure, the path can change to suit work requirements but must not stray outside the borders established by the declarative model:



To handle recurring work in a consistent way, a template would be created (e.g. rules for each purchase process in a declarative model.) A copy of that template is made for each work case (e.g.

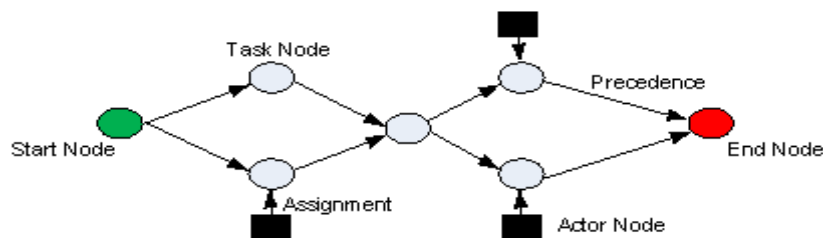
for each purchase the company makes.) The same declarative content is used to guide the building of the procedural content - each fulfilling its own needs:



### Procedural Models

Our procedural models are made up of the following node and link types:

- Task nodes represent activities. Tasks can be typed (an instance of a task type defined in the declarative model) or un-typed.
- Actor nodes represent people.
- Precedence links establish ordering amongst tasks. Multiple tasks can directly succeed one task. Many tasks can directly precede one task.
- Assignment links are used to associate actor nodes with tasks. A task can be associated with many actors. An actor can be associated with many tasks.
- Start and end nodes are used to encase the model. Start nodes have no incoming links. End nodes have no outgoing links.



### Declarative Models

Our declarative models are made up of the following nodes and links:

#### Task Types

It was shown how the procedural model can have tasks which are typed or un-typed. It is in the declarative model where task types are defined (for usage in the procedural model). As part of the task type definition, the number of times it can be instantiated is required.

For example, the task type 'select vendor' may be defined once in the declarative model. Multiple task instances of task type 'select vendor' may now be used in the procedural model.

#### Constraints

Task types form the parameters to the set of 5 constraints described below. Constraints can be:

- Simple - unconditional statements establish a target condition which must always hold in the procedural model.

- Complex - conditional statements establish a trigger and target condition. The target condition must hold in the procedural model if a trigger condition is present in the procedural model.

Task types form parameters in the role of a trigger or target. A trigger is established with a link from the type to the constraint. A target is created with a link from the constraint to the type (see examples.)

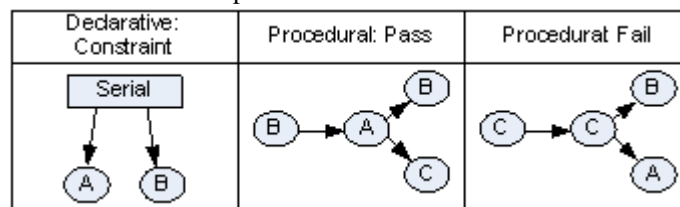
### Simple

The two following constraints are simple.

#### Serial

An unconditional constraint which can have multiple task type targets. It asserts that there should exist a path (either exclusively back or forward) from each instance of each target type to every other instance of each target type that exists in the procedural model.

Example: A and B are task types e.g. A is check delivery conditions, B is check payment conditions. The examples show a valid and invalid procedural resolution of that constraint.



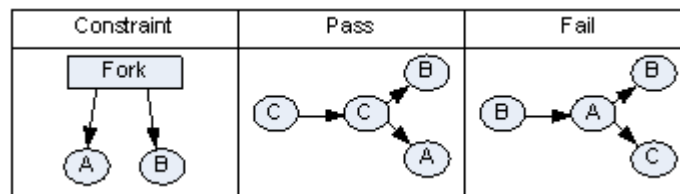
The second resolution has failed as there is no path from A to B.

**Essentially, look for a path (forward or backward) between any present instances of target types**

#### Fork

An unconditional constraint which can have multiple task type targets. It asserts that there should not exist a path (either exclusively back or forward) from each instance of each target type to every other instance of each target type that exists in the procedural model.

Example:



**Essentially, look for no path (forward or backward) between any present instances of target types**

### Complex

The following constraints are complex. For each of the following constraints, when multiple targets are specified, the user has the option to define the target mode as:

- each target type - (AND)
- one target type - (XOR)
- any target type - (OR)

For example, consider the include constraint. If an instance of a triggering task type is found, depending on the mode, it could imply include at least one of each of the targets types, it could imply include at least one of only one target type, or it could imply include at least one of any type.

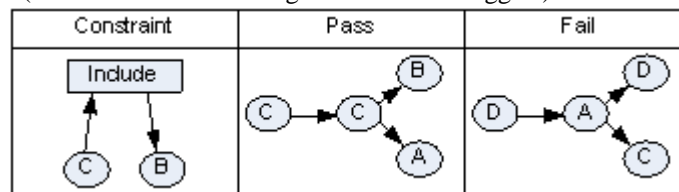


### Include

A conditional constraint which can have multiple task type triggers and targets. The constraint is triggered for each instance of a triggering type found anywhere in the procedural model. If triggered, the target condition must be present (anywhere in the model). For example, the trigger could be 'create purchase order' and the target could be 'log purchase order'.

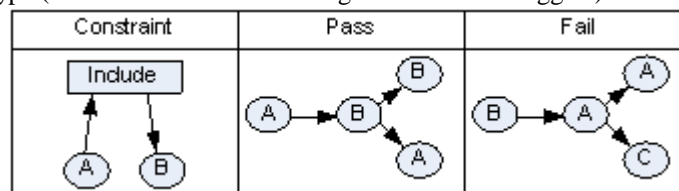
Options:

Weak - The same target instance may be used to satisfy the constraint for each instance of a triggering type (i.e. can use the same target for different triggers)



Pass as b can be used for both c's. Failed as a c is present but no b.

Strong - Different target instances must be used to satisfy the constraint for each instance of a triggering type (i.e. cannot use the same target for different triggers).



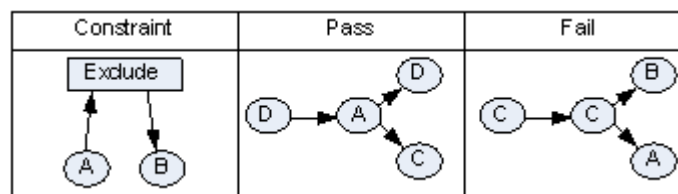
Pass as there is a b for each a. Failed as not enough b's present to match the a's .

**Essentially, if trigger present anywhere, look for target condition holding anywhere**

### Exclude

A conditional constraint which can have multiple task type triggers and targets. The constraint is triggered for each instance of a triggering type found anywhere in the procedural model. If triggered, the target condition must not be present (anywhere in the model). For example, A could be 'receive dispatch note' and B could be 'cancel order'.

Example



Pass as there is an a but no b's. Failed as there is an A but also a B present.

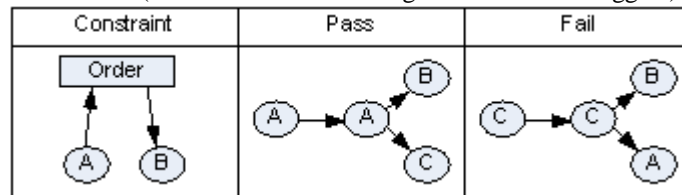
**Essentially, if trigger present anywhere, look for target condition not holding**

### Precedes (forward/with flow from trigger)

A conditional constraint which can have multiple task type triggers and targets. It asserts that in the forward path of any instance of a triggering type found in the procedural model, the target condition should hold. For example A could be 'amend order' and B could be 'receive confirmation'.

Options

Weak - Somewhere in the forward path of each instance of a triggering type, the target condition is fulfilled (i.e. can use the same targets for different triggers)



Pass as a b upstream of each A. Failed as no B upstream of A.

Strong - Somewhere in the forward path of each instance of a triggering type, the target condition is uniquely fulfilled (i.e. can not use the same targets for different triggers)

Chain - By selecting the chain option, the target condition must be found in the immediate forward path of the triggering instance

**Essentially, if trigger present anywhere, look for target condition in the forward path of the trigger**

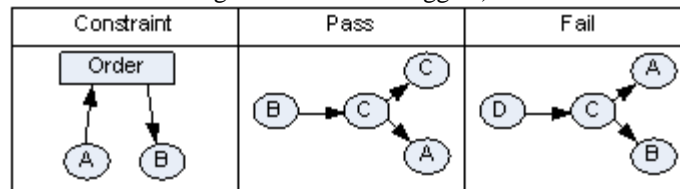
#### Succeeds (backward/against flow from trigger)

A conditional constraint which can have multiple task type triggers and targets. It asserts that in the backward path of any instance of a triggering type found in the procedural model, the target condition should hold. For example, A could be 'receive order confirmation' and B could be 'submit order'.

Options

Strong - Somewhere in the backward path of each triggering instance, the target condition is uniquely fulfilled

Weak - Somewhere in the backward path of each triggering instance, the target condition is fulfilled (i.e. can use the same targets for different triggers)



Pass as a B downstream of A. Failed as no b downstream of A.

Chain - By selecting the chain option, the target condition must be found in the immediate backward path of the triggering instance

**Essentially, if trigger present anywhere, look for target condition in the backward path of the trigger**

## **D2P Help - Part 2**

This document is intended to give an overview of D2P's user interface. For greater insight into the underlying techniques, see Help Part 1.

### **Contents**

[Introduction](#)

[Log into PowerMeeting](#)

[Create a D2P Model](#)

[The D2P Model Editor](#)

[Create a Node](#)

[Create a Link](#)

[Create a Declarative Model](#)

[Create a Procedural Model](#)

[Enacting a procedural Model](#)

[Using the verification tool](#)

[Exercise](#)

### **Introduction**

PowerMeeting is a web-based hub providing services for various collaborative processes. Applications can be 'plugged in' to PowerMeeting to leverage its web presence and group-working facilities. Each application is designed for some business, social or academic purpose where users collaboratively create, adapt, view and analyse graphical content. Examples of currently available applications include brainstorming, categorisation, presentations and calendars.

This document describes one such application. D2P is an approach to process support which balances scope for user flexibility with process compliance. A typical use maybe managing the treatment of patients where the treatment process is planned and executed.

### **Log into PowerMeeting**

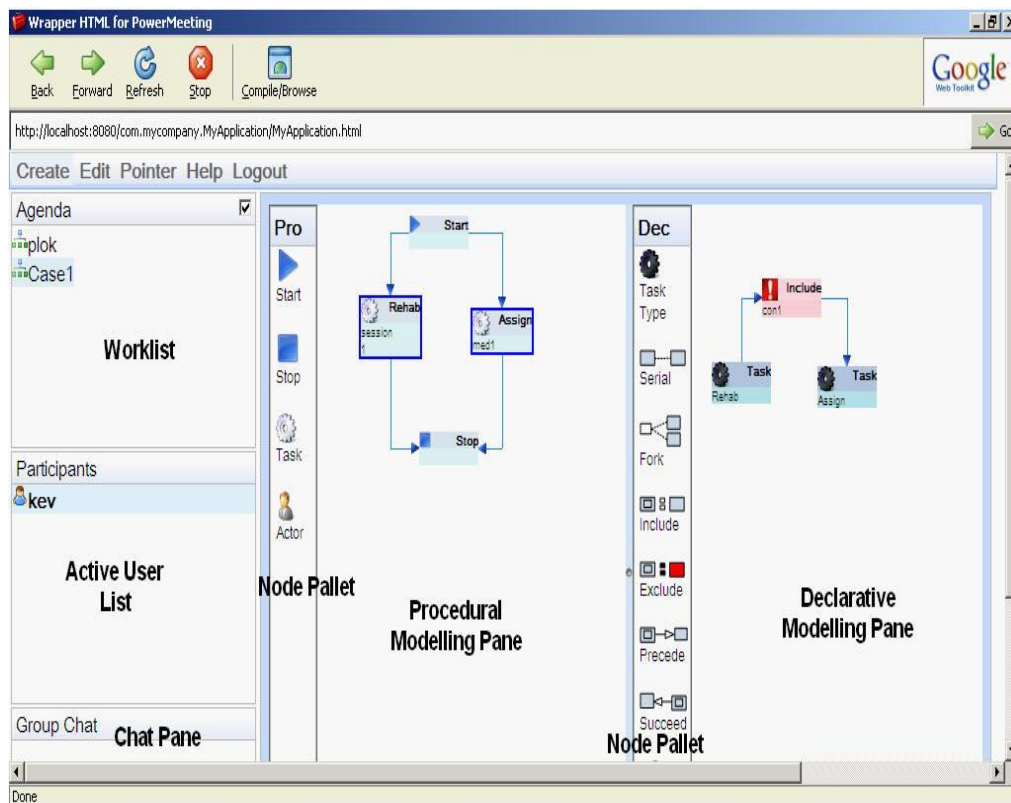
- 1 - Use an internet browser to access: <http://pc-364598-007.admbs.mbs.ac.uk/org.commonground.PowerMeeting/PowerMeeting.html>
- 2 - Enter a username. You will be known by this name for the duration of your session
- 3 - Enter a new or existing room name. Any work created in PowerMeeting will be associated with this room name.

### **Create a D2P model**

- 1/Select a template (predefined content) or create a new document (no predefined content):
  - Instantiate a template: create - template - 'desired template'
  - Create a new model: create - new - D2P
- 2/Name the new D2P document
- 3/A link to the new D2P document now appears in the document/work-list browser. Click the link to open the document.

### **D2P Model Editor**

Each D2P document is opened with a D2P document editor.



### Create a Node

- 1/Select a node type from either the procedural or declarative the pallet.
- 2/Fill in the create dialog; node types require at least a name and usually some additional parameters.
- 3/The new node will appear in the upper left corner on the relevant modelling pane.

Each node has the same underlying design.

- Click, hold and drag the upper left to move any node .
- For nodes in a declarative model, click the upper right to view further information. The information popup for constraint nodes is split into three tabs; description, triggers/targets and current state:



- Click the lower left of any node to show the menu. The main menu operations are:
  - Delete - removes the node and any associated links
  - Edit - edit properties of the node
  - Link - set the current node as a source for a new link (a link will be created when a target has been set)
  - Anchor - set the current node as a target for a link (a link will be created between the current source and target)

- Launch verify view - a verification view for the whole model

### Creating and deleting links

- 1/Set the source of the link by selecting the 'Link' option on the menu of any node
- 2/Set the target of the link by selecting the 'Anchor' option on the menu of any node
- 3/If the source and target are valid, a line will connect the nodes

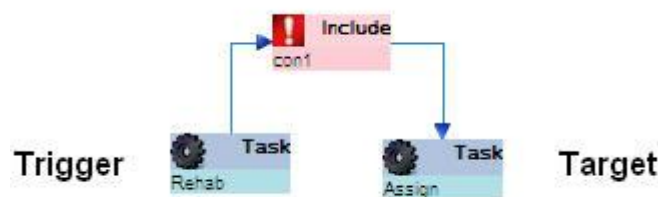
By hovering over a link, the user is given the option to delete that link.


### Create a Declarative Model


Declarative models are used to establish constraints which must hold in the procedural model e.g. task 'assign medication' can be performed many times in a single case. If 'assign medication' occurs then at some point at least one 'rehabilitation session' must also have occurred. Let us encode this constraint:

- 1/Create task types: From the description above we can extract two task types; assign medication and rehabilitation session. A task type would be created for each.
- 2/Create constraints: From the description above we can assume that the include pattern is the most suitable, as no ordering requirements are mentioned. We create an include constraint.
- 3/Connect task types: Task types are used as parameters to form useful constraints. In the above case, the assign medication task type forms the trigger and the rehabilitation session task type forms the target. Triggers flow into a constraint node. Targets flow out of a constraint node.

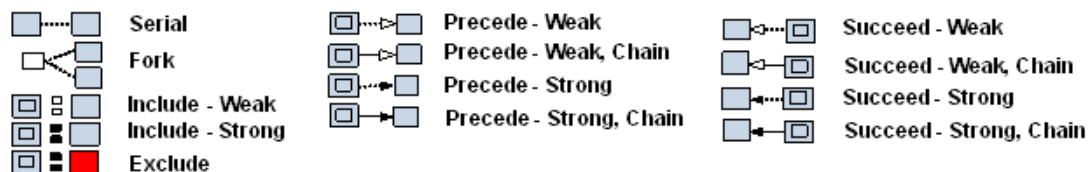
The below constraint establishes that, in the procedural model, if an instance of 'rehab' is used, then an instance of 'assign' should also be used.



Generic constraint symbol on nodes: 

Task type symbol: 

Detailed key for constraint symbols, found on pop-up on constraint nodes and declarative modelling pallet:



### Create a Procedural Model

For each work case a procedural model is created which satisfies both the needs of the case and constraints of the declarative model.

1/ Create tasks: select task from the pallet. To assign this task instance a type, select the relevant type from the drop-down list.

2/ Create actor: select actor from the pallet. Currently the only parameter required for each actor is a name.

To form a more structured procedural model tasks can be connected to establish precedence and actors and can be connected to tasks to establish allocation.

Task instance symbol:  Actor symbol:  Start symbol:  End symbol: 

### Enacting a Procedural Model

The state of each task is indicated by the colour of its border: blue for waiting, yellow for ready, black for ongoing, green for completed and red for cancelled. Each task node implements a state machine mechanism; waiting to ready, ready to ongoing, ongoing to completed and cancelled which can be reached from any state except completed.

States can be manually set through the 'set state' menu operation of the relevant task node. Where relevant structure is in place, task states are automatically set.

### Using the verification tool

Verification establishes the current state of each declared constraint by analysing the current procedural resolution. A constraint can be in one of five states:

Black – constraint has not been triggered (e.g. when no instance of 'rehab' is used in the procedural model)

Yellow – constraint satisfied if model executed in current state

Green – constraint is satisfied.

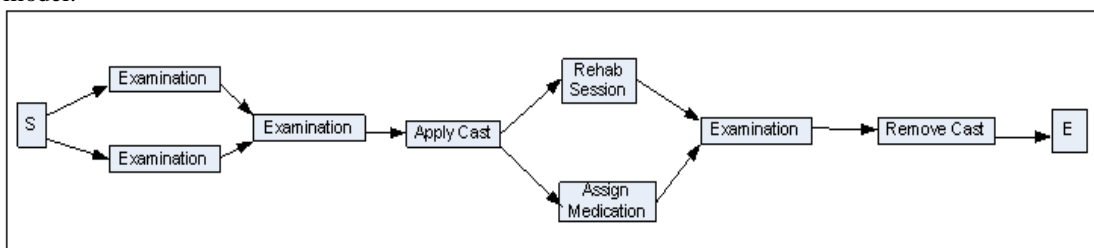
Pink – constraint not satisfied. This error is correctable as relevant parts of the model have not yet been executed

Red - constraint not satisfied. This error is not correctable as relevant parts of the model have been executed

The current state of an individual constraint can be checked with the individual popup (see previous section). To check all constraints, a verification view can be launched via menu operation on any node or from the tool pallet.

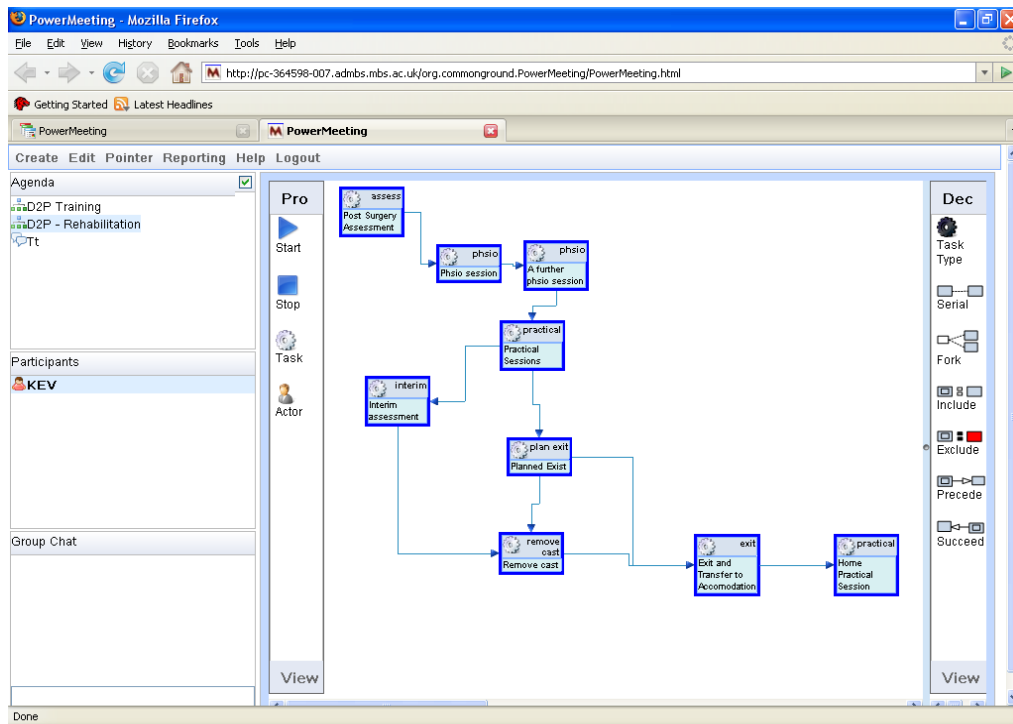
### Exercise

Let us further develop our scenario. A template is available in PowerMeeting system called 'Training Task'. This template is a brief declarative model for our treatment of a fractured arm. Instantiate the template and create a procedural which fulfils it. Pictured below is just one possible way the declarative model could be resolved. Get a feel for the system by building your own procedural model.

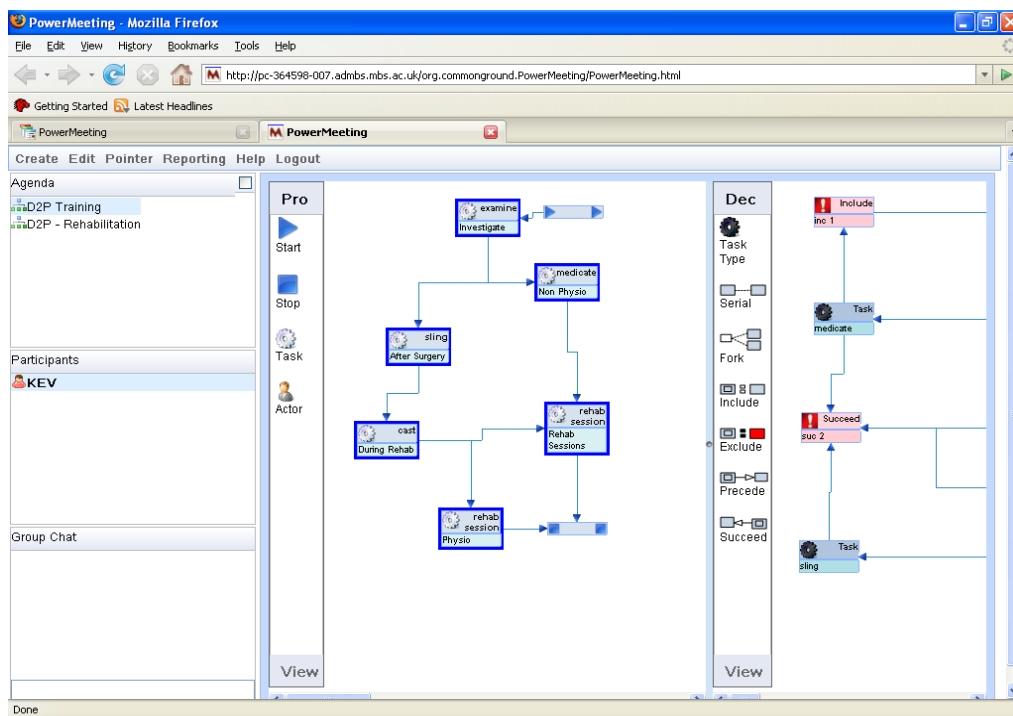




## Task model from room T2:

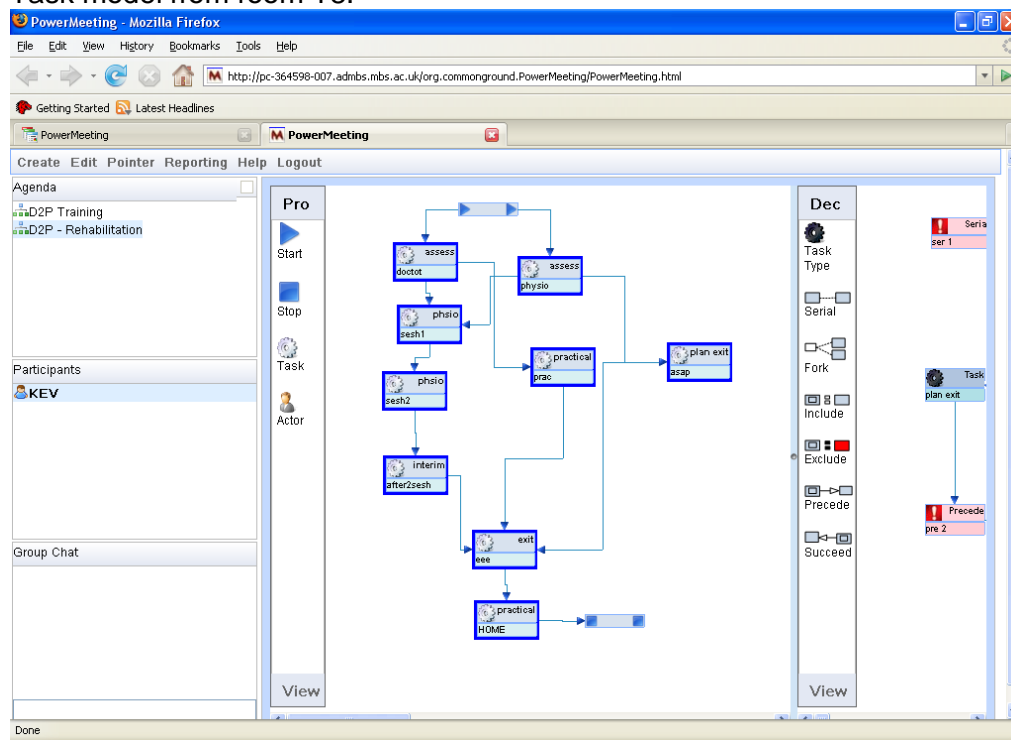


## Training Model from room T3:





## Task model from room T8:



# Appendix J

## Usability Study Results

Task success for individuals:

A-T represents the 20 qualities of the marking scheme (presented in Section 5.4).

| R<br>oo<br>m | A   | B           | C   | D   | E | F   | G | H | I | J   | K   | L   | M   | N | O   | P | Q   | R   | S      | T  | Total         |
|--------------|-----|-------------|-----|-----|---|-----|---|---|---|-----|-----|-----|-----|---|-----|---|-----|-----|--------|----|---------------|
| T<br>7       | 0.8 | 1           | 0.6 | 1   | 1 | 0.6 | 1 | 1 | 1 | 0.6 | 0.6 | 1   | 1   | 1 | 1   | 1 | 1   | 1   | -      | -2 | 14.2<br>(79%) |
| T<br>8       | 0.8 | 0<br>.<br>8 | 1   | 1   | 1 | 1   | 1 | 0 | 1 | 1   | 1   | 1   | 1   | 1 | 1   | 0 | 0.6 | 0.8 | -<br>2 | -  | 14<br>(78%)   |
| T<br>9       | 0   | 0           | 0.8 | 0.8 | 1 | 0.8 | 1 | 1 | 1 | 0   | 0   | 0.6 | 0.6 | 1 | 0   | 1 | 0.6 | 0.8 | -      | -  | 11<br>(61%)   |
| T<br>10      | 0.6 | 0<br>.<br>6 | 0.6 | 1   | 1 | 1   | 1 | 1 | 1 | 0.6 | 1   | 0.6 | 0.6 | 1 | 0.6 | 1 | 1   | 1   | -      | -  | 15.2<br>(85%) |
| T<br>11      | 1   | 1           | 0.8 | 1   | 1 | 1   | 1 | 1 | 1 | 1   | 1   | 0.6 | 0.6 | 1 | 1   | 1 | 1   | 0.8 | -      | -  | 16.8<br>(93%) |
| T<br>13      | 0   | 0           | 0.6 | 0.6 | 1 | 0.6 | 1 | 1 | 1 | 0   | 0   | 1   | 1   | 1 | 0   | 1 | 0.6 | 0.6 | -      | -  | 11<br>(61%)   |

Task success for groups

A-T represents the 20 qualities of the marking scheme (presented in Section 5.4).

| Room | A | B | C   | D   | E | F | G | H | I | J | K | L   | M   | N | O | P   | Q | R   | S      | T | Total<br>(Max<br>18) |
|------|---|---|-----|-----|---|---|---|---|---|---|---|-----|-----|---|---|-----|---|-----|--------|---|----------------------|
| T1   | 1 | 0 | 0.6 | 0.6 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0.6 | 1   | 1 | 1 | 0.8 | 1 | 0.8 | -<br>1 | - | 12.4<br>(69%)        |
| T2   | 0 | 0 | 0.6 | 1   | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0.6 | 1   | 1 | 1 | 1   | 1 | 1   | -<br>1 | - | 13.2<br>(73%)        |
| T3   | 1 | 0 | 1   | 1   | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0.6 | 0.6 | 1 | 1 | 1   | 1 | 1   | -      | - | 16.2<br>(90%)        |

### Individual Questionnaire result

| Question | T7 | T8 | T9 | T10 | T11 | T13 | Mean | Standard Deviation |
|----------|----|----|----|-----|-----|-----|------|--------------------|
| 1        | 3  | 5  | 4  | 5   | 5   | 4   | 4.5  | 0.8                |
| 2        | 4  | 3  | 3  | 4   | 5   | 5   | 4    | 0.9                |
| 3        | 4  | 4  | 5  | 5   | 5   | 5   | 5    | 0.5                |
| 4        | 4  | 3  | 3  | 3   | 4   | 5   | 3.5  | 0.8                |
| 5        | 4  | 4  | 5  | 5   | 4   | 5   | 4.5  | 0.5                |
| 6        | 5  | 3  | 4  | 4   | 4   | 4   | 4    | 0.6                |
| 7        | 4  | 4  | 5  | 5   | 4   | 5   | 4.5  | 0.5                |
| 8        | 3  | 3  | 4  | 3   | 4   | 5   | 3.5  | 0.8                |
| 9        | 4  | 4  | 4  | 4   | 5   | 5   | 4    | 0.5                |
| 10       | 5  | 4  | 4  | 4   | 5   | 5   | 4.5  | 0.5                |
| 11       | 5  | 4  | 4  | 4   | 4   | 5   | 4    | 0.5                |
| 12       | 4  | 3  | 4  | 3   | 4   | 4   | 4    | 0.5                |

### Group Questionnaire result

| Question | T1 | T2 | T3 | T4 | T5 | T6 | Mean | Standard Deviation |
|----------|----|----|----|----|----|----|------|--------------------|
| 1        | 4  | 5  | 4  | 4  | 5  | 4  | 4    | 0.5                |
| 2        | 5  | 3  | 4  | 4  | 4  | 3  | 4    | 0.8                |
| 3        | 5  | 4  | 4  | 5  | 5  | 3  | 4.5  | 0.8                |
| 4        | 5  | 3  | 3  | 3  | 4  | 3  | 3    | 0.8                |
| 5        | 5  | 4  | 3  | 4  | 5  | 3  | 4    | 0.9                |
| 6        | 4  | 5  | 5  | 4  | 5  | 3  | 4.5  | 0.8                |
| 7        | 4  | 5  | 5  | 4  | 5  | 4  | 4.5  | 0.5                |
| 8        | 4  | 3  | 4  | 4  | 5  | 3  | 4    | 0.8                |
| 9        | 4  | 4  | 4  | 4  | 5  | 3  | 4    | 0.6                |
| 10       | 5  | 5  | 4  | 5  | 5  | 4  | 5    | 0.5                |
| 11       | 4  | 5  | 3  | 3  | 4  | 5  | 4    | 0.9                |
| 12       | 4  | 4  | 3  | 4  | 5  | 5  | 4    | 0.8                |
| 13       | 5  | 5  | 5  | 5  | 4  | 5  | 5    | 0.4                |
| 14       | 5  | 4  | 5  | 5  | 4  | 3  | 4.5  | 0.4                |
| 15       | 3  | 3  | 3  | 5  | 4  | 4  | 3.5  | 0.8                |

## References

- Aalst, W. M. P. v. d., M. Adams, et al. (2009). Flexibility as a Service. DASFAA 2009 Workshops, LNCS 5667, Springer-Verlag Berlin Heidelberg.
- Aalst, W. M. P. v. d., L. Aldred, et al. (2004). Design and Implementation of the YAWL System. Advanced Information Systems Engineering, Proceedings of the 16th International Conference on Advanced Information Systems Engineering (CAiSE'04), volume 3084 of Lecture Notes in Computer Science, Springer-Verlag, Berlin.
- Aalst, W. M. P. v. d., A. H. M. t. Hofstede, et al. (2003). Workflow Patterns. Distributed and Parallel Databases 14(1): 5-51.
- Aalst, W. M. P. v. d., A. H. M. t. Hofstede, et al. (2003). Business Process Management: A Survey. Business Process Management, Springer-Verlag Berlin Heidelberg.
- Aalst, W. M. P. v. d. and M. Pesic (2006). DecSerFlow: Towards a Truly Declarative Service Flow Language. WS-FM 2006. LNCS, Springer, Heidelberg.
- Aalst, W. M. P. v. d. and M. Pesic (2006). Specifying, Discovering and Monitoring Service Flows: Making Web Services Process-aware. BPM Center Report BPM-06-09, BPM Center.
- Aalst, W. v. d. (1998). The Application of Petri Nets to Workflow Management. Journal of Circuits, Systems and Computers, 8(1): 21-66.
- Adams, M. J. (2007). Facilitating Dynamic Flexibility and Exception Handling for Workflows. Faculty of Information Technology. Brisbane, Australia, Queensland University of Technology.
- Agarwal, R., P. De, et al. (1999). Comprehending Object and Process Models: An Empirical Study. IEEE Transactions on Software Engineering 25(4).
- Agostini, A. and G. D. Michelis. (2000). A Light Workflow Management System Using Simple Process Models. Computer Supported Cooperative Work 9(3/4): 335-363.
- Alexander, C. (1979). The Timeless Way of Building. Oxford, Oxford University Press.
- Bajec, M. and M. Krisper (2005). A Methodology and Tool Support for Managing Business Rules in Organisations. Information Systems 30.
- Baker, K., S. Greenberg, et al. (2002). Empirical Development of a Heuristic Evaluation Methodology for Shared Workspace Groupware. CSCW. New Orleans, Louisiana, USA, ACM.
- Bansler, J. P. and K. Bødker (1993). A Reappraisal of Structured Analysis: Design in an Organizational Context. ACM Transactions on Information Systems (TOIS 11(2): 165-193.

- Bernstein, A. (2000). How can Cooperative Work Tools Support Dynamic Group Processes? Bridging the Specificity Front. *Computer Supported Cooperative Work*: 279-288.
- Caplinskas, A. and O. Vasilecas (2004). Information systems research methodologies and models. *Proceedings of the 5th international conference on Computer systems and technologies*, Bulgaria, ACM New York (NY).
- Cardoso, J. (2006). Process Control-flow Complexity Metric: An Empirical Validation. *IEEE International Conference on Services Computing (IEEE SCC 06)*. Chicago, USA, IEEE Computer Society.
- Chiu, D., Q. Li, et al. (2000). A Logical Framework for Exception Handling in ADOME Workflow Management System. In *Proceedings of the 12th International Conference on Advanced Information Systems Engineering (CAiSE 2000)*. Stockholm, Sweden, volume 1789 of *Lecture Notes in Computer Science*.
- Collins, J., W. Ketter, et al. (2010). Flexible Decision Support in Dynamic Inter-organisational Networks. *European Journal of Information Systems* 19(4): 436-448.
- Courtney, J. (2001). Decision Making and Knowledge Management in inquiring Organizations: Toward a new Decision-Making Paradigm for DSS. *Decision Support Systems* 31: 17-38.
- Cugola, G. (1998). Tolerating Deviations in Process Support Systems via Flexible Enactment of Process Models. *IEEE Transactions on Software Engineering* 24(11): 982-1001.
- Daoudi, F. and S. Nurcan (2007). A Benchmarking Framework for Methods to Design Flexible Business Processes. *Software Process Improvement and Practice* 12: 51-63.
- Davis, F. D. (1989). Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. *MIS Quarterly* 13(3): 319-340.
- Davies, I., P. Green, et al. (2006). How do Practitioners use Conceptual Modelling in Practice? *Data & Knowledge Engineering* 58: 358-380.
- Debevoise, T. (2005). *Business Process Management with a Business Rules Approach*. VA, USA, Business Knowledge Architects, Roanoke.
- Detmar, S., D. Gefen, et al. (2007). The ISWorld Quantitative, Positivist Research Methods Website. Retrieved 2008, from <http://www.dstraub.cis.gsu.edu:88/quant/>.
- Dix, A., J. Finley, et al. (2004). *Human Computer Interaction*, 3rd Edition, Pearson-PrenticeHall.
- Dourish, P., J. Holmes, et al. (1996). Freeflow: Mediating Between Representation and Action in Workflow Systems. *ACM CSCW Conference*. Boston, USA.
- Dumas, M., W. M. v. d. Aalst, et al. (2005). *Process Aware Information Systems: Bridging People and Software through Process Technology*, Wiley-Interscience.

- Ellis, C., K. Keddara, et al. (1995). Dynamic Change within Workflow Systems. ACM Conference on Organizational Computing Systems (COOCS). Milpitas, CA, USA.
- Engelbart, D. C. (1992). Toward High-Performance Organizations: A Strategic Role for Groupware. Groupware 92, California, Morgan Kauffman.
- EXTERNAL (2003). DNV Public Web for EXTERNAL, from [www.external-ist.org](http://www.external-ist.org).
- Facebook (2010). [www.facebook.com](http://www.facebook.com).
- Fahland, D., D. Lubke, et al. (2009). Declarative versus Imperative Process Modelling Languages: The Issue of Understandability. BPMDS 2009 and EMMSAD 2009, LNBIP 29, Springer-Verlag Berlin Heidelberg.
- Faustmann, G. (2000). Configuration for Adaptation - A Human-centred Approach to Flexible Workflow Enhancement. Computer Supported Cooperative Work 9: 413-434.
- Feldkamp, D., K. Hinkelmann, et al. (2007). KISS – Knowledge-Intensive Service Support: An Approach for Agile Process Management. RuleML 2007, LNCS 4824, Springer-Verlag Berlin Heidelberg.
- Fischer, G., E. Giaccardi, et al. (2004). Meta-design: A Manifesto for End-user Development. Communications of the ACM 24(9): 33-37.
- FlashMeeting. (2010). "<http://flashmeeting.open.ac.uk/>."
- Galbraith, J. R. (2001) Designing Organizations: An executive guide to Strategy, Structure and process. Jossey Bass Wiley, second edition.
- Garrett, J. J. (2005). Ajax: A New Approach to Web Applications. Adaptive Path.
- Gartner (2003). The Real-time Enterprise, [gartner.com/rte](http://gartner.com/rte).
- Gemino, A. and Y. Wand (2004). A Framework for Empirical Evaluation of Conceptual Modeling Techniques. Requirements Eng 9: 248-260.
- Georgakopoulos, D., M. Hornick, et al. (1995). An Overview of Workflow Management: From Process Modelling to Workflow Automation Infrastructure. Distributed and Parallel Databases 3.
- Glance, N. S., D. S. Pagani, et al. (1996). Generalized Process Structure Grammars (GPSG) for Flexible Representation of Work. ACM CSCW Conference. Boston USA.
- Goedertier, S., R. Haesen, et al. (2007). EMBrA2CE: A Vocabulary and Execution Model for Declarative Process Models. FETEW research report, K.U.Leuven
- Gonzalez, E., Smari, W. W. and Debnath, N (1999). Towards Flexibility of Workflow Management Systems Based on Task Requirement Classification. Simulation series: Society for computer simulation 34(2): 210-222.
- Google (2010). "<http://wave.google.com/about.html>."

- Gottschalk, F., W. M. P. V. D. Aalst, et al. (2008). Configurable Workflow Models. *International Journal of Cooperative Information Systems* 18(5).
- Greenwood, R. M., I. Robertson, et al. (1995). Active Models in Business. 5th Conference on Business Information Technology.
- Greiner, U., J. Ramsch, et al. (2004). Adaptive Guideline-based Treatment Workflows with Adaptflow. *Proceedings of the Symposium on Computerized Guidelines and Protocols Prague*, IOS Press.
- Hagen, C. and G. Alonso (2000). Exception Handling in Workflow Management Systems." *IEEE Transactions on Software Engineering* 26(10): 943-958.
- Hamlet, D. and J. Maybee (2001). *The Engineering of Software*, Addison Wesley.
- Hammer, M. and J. Champy (2001). *Reengineering the Corporation: A Manifesto for Business Revolution*. New York (NY), Nicholas Brealey Publishing; 3rd Revised edition
- Hevner, A., S. T. March, et al. (2004). Design Science in Information Systems Research. *MIS Quarterly* 28(1): 75-105.
- IBM (2010). "<http://www.lombardisoftware.com/>."
- ISO 9241-11 (1998). International Standards Organisation; ISO 9241-11 – Ergonomic Requirements for Office Work with Visual Display Terminals (VDT's; Part 11 – Guidance on usability, 22).
- Jablonski, S. (1994). MOBILE: A Modular Workflow Model and Architecture. Conference on Dynamic Modelling and Information Systems. Noordijkerhout, Netherlands.
- Johnson, B. (2006). GWT: What, Why, and How, JavaZone. JavaZone.
- Jørgensen, H. D. (2002). Interactive Process Models for Knowledge Intensive Project Work. CAiSE Doctoral Consortium, Toronto, Canada.
- Jørgensen, H. D. (2004). Interactive Process Models. Department of Computer and Information Science. Trondheim, Norway, Norwegian University of Science and Technology. PhD.
- Kammer, P. J., G. A. Bolcher, et al. (2000). Techniques for Supporting Dynamic and Adaptive Workflow. *Computer Supported Cooperative Work* 9: 269-292.
- Kardasisa, P. and P. Loucopoulos (2004). Expressing and Organising Business Rules. *Information and Software Technology* 46: 701-718.
- Klein, M., C. Dellarocas, et al. (2000). Introduction to the Special Issue on Adaptive Workflow Systems. *Computer Supported Cooperative Work* 9: 265-267.
- Klöckner, K. (2002). BSCW. Cooperation Support for Distributed Workgroups. In: 10th Euromicro Workshop on Parallel, Distributed and Network-based Processing, IEEE Computer Society.

- Krogstie, J. (2007). Modelling of the People, by the People, for the People. *Conceptual Modelling In Information Systems Engineering*: 305-318.
- Krogstie, J. (2008). Using EEML for Combined Goal and Process Oriented Modelling: A Case Study. *Proceedings of EMMSAD*.
- Krogstie, J. and H. D. Jørgensen (2004). *Interactive Models for Supporting Networked Organisations*. CAiSE, LNCS 3084, Springer-Verlag Berlin Heidelberg.
- Krogstie, J., G. Sindre, et al. (2006). Process Models Representing Knowledge for Action: A Revised Quality Framework. *European Journal of Information Systems* 15: 91-102.
- Kueng, P. and P. Kawalek (1997). Goal based process models: creation and evaluation. *Business Process Management Journal* 3(1): 17-38.
- Kuntz, J. C., T. R. Christiansen, et al. (1998). The Virtual Design Team: A Computational Simulation Model of Project Organizations. *Communications of the ACM* 41(11).
- Kwan, M. M. and P. Balasubramanian (2003). KnowledgeScope: Managing Knowledge in Context. *Decision Support Systems* 35: 467-486.
- Leuf, B. and W. Cunningham (2001). *The Wiki Way*, Addison Wesley.
- Levitt, B. and March J. G. (1998). Organizational Learning. *Annual Review of Sociology* (1998) Vol.14, pp. 319-340.
- Lewis, C. H. and G. M. Olson (1989). *Can Principles of Cognition Lower the Barriers to Programming? Empirical Studies of Programmers: Second Workshop*, Norwood, NJ, Ablex Publishing.
- Lillehagen, F. (1999). Visual Extended Enterprise Engineering Embedding Knowledge Management, Systems Engineering and Work Execution. *IFIP International Enterprise Modelling Conference*. Vardal, Norway.
- Lillehagen, F. (2004). The Foundation of the AKM Technology. *Proceedings of CE, Madeira*.
- Lillehagen, F., J. Krogstie, et al. (2002). Active Knowledge Models for Supporting eWork and eBusiness. *International Conference on Concurrent Enterprising (ICE)*. Rome, Italy.
- Lillehagen, F., W. Wang, et al. (2004). The EXTERNAL Experience on System and Enterprise Integration. *MIS 2003*.
- Liu, L. and C. Pu (1997). ActivityFlow: Towards Incremental Specification and Flexible Coordination of Workflow Activities. *Proceedings of the 16th International Conference on Conceptual Modelling (ER'97)*, Los Angeles, California.
- Ly, L. T., S. Rinderle, et al. (2006). Semantic Correctness in Adaptive Process Management Systems. *Proceedings of the 4th International Conference on*



- Business Process Management (BPM'06), volume 4102 of Lecture Notes in Computer Science, Vienna, Austria, Springer Verlag.
- Markus, M. L., A. Majchrzak, et al. (2002). A Design Theory for Systems That Support Emergent Knowledge Processes. *MIS Quarterly* 26(3): 179-212.
- Marshall, C. C. and F. M. S. III (1997). Spatial Hypertext and the Practice of Information Triage. *Proceedings of ACM Hypertext '97*.
- Mason, R. O. and I. I. Mitroff (1973). A Program for Research on Management Information Systems. *Management Science* 19(5).
- Microsoft. (2010). "<http://get.live.com/messenger/overview>."
- Miers, D. and R. Hunt (1995). *Process Product Watch - Work Management Technologies Report - Evaluation Framework Process Support Systems*. Enix, UK.
- Muehlen, M. Z. (2004). Organizational Management in Workflow Applications - Issues and Perspectives. *Information Technology and Management* 5: 271-291.
- Muehlen, M. z. and M. Rosemann (2005). Integrating Risks in Business Process Models. *16th Australasian Conference on Information Systems*. Sydney.
- Muller, R., U. Greiner, et al. (2004). AgentWork: a Workflow System Supporting Rule-based Workflow Adaptation. *Data & Knowledge Engineering* 51(2): 223-256.
- Nardi, B. A. and J. A. Johnson (1994). User Preferences for Task-specific vs. Generic Application Software. *Conference on Human Factors in Computing Systems*, Boston, Massachusetts, United States, ACM New York, NY, USA.
- Nielson, J. (1994). Estimating the Number of Subjects Needed for a Thinking Aloud Test. *International Journal of Human Computer Interaction* 41(3): 385-397.
- Nielson, J. (1994). *Heuristic Evaluation. Usability Inspection Methods*. John Wiley, New York 1994.
- Notes, L. (2010). "<http://www-142.ibm.com/software/swlotus/products/product4.nsf/wdocs/noteshomepage>."
- Nutt, G. J. (1996). The Evolution Towards Flexible Workflow Systems. *Distributed Systems Engineering* 3: 276-294.
- Oquendo, F., K. N. Papamichail, et al. (2000). Addressing Decision Making Issues in Enterprise Process Modelling. *Proc. Int Conf on Enterprise Information Systems*, American Mathematical Society.
- Oryx (2010). "<http://bpt.hpi.uni-potsdam.de/Oryx>."
- Owen, M. and J. Raj (2003). *BPMN and Business Process Management Introduction to the New Business Process Modelling Standard*. Popkin Software Deliverable.

- Perry, D. E., A. A. Porter, et al. (2000). Empirical Studies of Software Engineering: A Roadmap. International Conference on Software Engineering. Limerick, Ireland, ACM New York, NY, USA.
- Pesic, M. (2008). Constraint-Based Work Flow Management Systems: Shifting Control to Users. Technische Universiteit Eindhoven. Eindhoven.
- Pesic, M. and W. M. P. v. d. Aalst (2006). A Declarative Approach for Flexible Business Processes Management. BPM 2006 Workshops, Springer Verlag.
- Pesic, M. and W. M. P. van der Aalst (2006). A Declarative Approach for Flexible Business Processes. Eder, J., Dustdar, S. (eds.) Business Process Management Workshops . LNCS, vol. 4103, pp. 169-180. , Springer, Heidelberg.
- Powell, A., G. Piccoli, et al. (2004). Virtual Teams: A Review of Current Literature and Directions for Future Research. The DATA BASE for Advances in Information Systems 35(1): 6-36.
- ProcessWave (2010). "<http://www.processwave.org/>."
- Reijers, H. A. (2006). Workflow Flexibility: The Forlorn Promise. 15th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises. Manchester, United Kingdom, IEEE Computer Society.
- Rey, G. Á., L. Cekov, et al. (2008). D2.1.2 Service Modelling Tools Design. S0A4ALL Project Deliverable.
- Rosemann, M., J. Recker, et al. (2006). A Study of the Evolution of the Representational Capabilities of Process Modelling Grammars. CAiSE 2006, LNCS 4001, Springer-Verlag Berlin Heidelberg.
- Russell, N., A. H. M. t. Hofstede, et al. (2005). Workflow Resource Patterns: Identification, Representation and Tool Support. Advanced Information Systems Engineering In Lecture Notes in Computer Science 3520: 216-232.
- Russell, N., A. H. M. t. Hofstede, et al. (2004). Workflow Data Patterns. Technical Report QUT Technical Report FIT-TR-2004-01, Queensland University of Technology, Brisbane.
- Sadiq, S., G. Governatori, et al. (2007). Modelling Control Objectives for Business Process Compliance. Proceedings of the 5th international conference on Business process management, Australia, Lecture Notes in Computer Science, Springer-Verlag Berlin Heidelberg
- Sadiq, S. W., M. E. Orlowska, et al. (2005). Specification and Validation of Process Constraints for Flexible Workflows. Information Systems 30(5): 349-378.
- SAP (2010). "<http://www.sdn.sap.com/irj/scn/weblogs?blog=/pub/wlg/15618>."
- Sarin, S. (1995). Flexible Workflow Architectures: The New Adaptive Workflow. Workflow Conference. San-Jose, USA.

- Savolainen, T., D. Beeckmann, et al. (1995). Positioning of Modelling Approaches, Methods and Tools. *Comput Ind* 25: 255-262.
- Scheer, A. W. (2000). *ARIS - Business Process Modelling*, Springer, Berlin.
- Schmidt, K. (1999). Of maps and scripts: The Status of Formal Constructs in Cooperative Work. *Information and Software Technology* 41: 319-329.
- Schonenberg, M. H., R. S. Mans, et al. (2006). Taxonomy of process flexibility. Technical Report BPM Centre Report BPM-06-22.
- Schummer, T. and S. Lukosch (2007). *Patterns for Computer-Mediation Interaction*, Wiley.
- Selvin, A. M. and S. J. B. Shu (2002). Rapid Knowledge Construction: A Case Study in Corporate Contingency Planning Using Collaborative Hypermedia. *Knowledge and Process Management* 9(2): 119-128.
- Shum, S. J. B., A. M. Selvin, et al. (2006). Hypermedia Support for Argumentation-Based Rationale: 15 Years on from gIBIS and QOC. *Rationale Management in Software Engineering* 1: 111-132.
- Siau, K. and M. Rossi (1998). Evaluation of Information Modelling Methods - A Review. *Thirty-First Annual Hawaii International Conference on System Sciences*. Kohala Coast, HI.
- Simon H. A. (1996). *The Sciences of the Artificial*, Second edition. The MIT Press Cambridge, Massachusetts London England.
- Skype. (2010). "<http://www.skype.com/>."
- soa4all. (2010). "<http://www.soa4all.eu/>."
- Suchman, L. (1995). Representation of Work - Introduction to Special Issue. *Communications of the ACM* 38(9).
- Suchman, L. A. (1987). *Plans and Situated Actions: The Problem of Human Machine Communication*. Cambridge, United Kingdom:, Cambridge University Press.
- Szajna, B. (1996). Empirical Evaluation of the Revised Technology Acceptance Model. *Management Science* 42(1): 85-92.
- Tullis, T. and B. Albert (2008). *Measuring the User Experience*, Morgan Kaufmann.
- UML 2010. OMG UML Specification, Object Management Group, <http://www.uml.org/>
- Vaishnavi, V and Kuechler, B. (2005). *Design Research in Information Systems*, Association for Information Systems, 2005.
- Wainer, J., F. Bezerra, et al. (2004). Tucupi: A Flexible Workflow System Based on Overridable Constraints. *2004 ACM Symposium on Applied Computing*. Nicosia, Cyprus, ACM.

- Wand, Y. and R. Weber (2002). Research Commentary: Information Systems and Conceptual Modelling - A Research Agenda. *Journal of Information Systems Research* 13(4): 363-376.
- Wang, W. (2003). Visualizing and Interacting with Hypermedia-based Process-centric Enterprise Models. *Journal of Network and Computer Applications* 26: 73-93.
- Wang, W. (2008). Incorporating Computational Semantics into Emergent Hypermedia Structures. *International Journal of Web Engineering and Technology* 4(1): 4-20.
- Wang, W. (2008). PowerMeeting on CommonGround: Web Based Synchronous Groupware with Rich User Experience. In *Proceedings of the Hypertext 2008, Workshop on Collaboration and Collective intelligence*. Pittsburgh, PA, USA, ACM, New York, NY.
- Wang, W., K. Finch, et al. (2009). A Cooperative Hypermedia Approach to Flexible Process Support for Managing Distributed Projects. *International Journal of Cooperative Information Systems* 18(3-4): 418-512.
- Wang, W., J. M. Haake, et al. (2004). Supporting Virtual Meetings in the Overall Business Context. *International Journal of Computer Applications in Technology* 19(3/4): 195-208.
- Weber, B., M. Reichert, et al. (2008). Change Patterns and Change Support Features - Enhancing Flexibility in Process-aware Information Systems. *Data and Knowledge Engineering*: 438-466.
- Weber, B., H. A. Reijers, et al. (2009). The Declarative Approach to Business Process Execution: An Empirical Test. *CAiSE 2009, LNCS 5565*, Springer-Verlag Berlin Heidelberg.
- Weber, B., S. Sadiq, et al. (2009). Beyond Rigidity - Dynamic Process Lifecycle Support: A Survey on Dynamic Changes in Process-aware Information Systems." *COMPUTER SCIENCE - RESEARCH AND DEVELOPMENT* 23(2): 47-65.
- WfMC (2001). *Workflow Management Coalition, WfMC Workflow Handbook* Florida, USA, Future Strategies Inc.
- Zelkowitz, M. V. and D. Wallace (1998). "Experimental Models For Validating Computer Technology." *IEEE Computer* 31(5): 23-31.