A Study on Process Simulate Software & Modelling

Kumar H T, Dr. K R Prakash
M. Tech in Industrial Automation and Robotics, Student of NIE College Mysuru, India
Head of Department of Mechanical Engineering, The National Institute of Engineering Mysuru

***Abstract*—A digital manufacturing system for 3D manufacturing process verification is called Process Simulate. By enabling manufacturing firms to digitally evaluate manufacturing concepts up front and throughout the lifespan ofnew product introductions, Process Simulate is a key speed-to- market facilitator. Utilizing 3D data of resources and products makes it easier to virtually validate, optimize, and commission complicated manufacturing processes, leading to a quicker start-up and greater production quality. The tactical oversight of software development, supporting process changes, and software project leadership training are all areas where software simulation modeling is being used more and more. Applications for software process simulation span the life cycle, from short-term targeted stages to longer-term product evolutionary models have significant organizational effects. An overview of the research being done in this field is given in this article. It specifies the inquiries.**

I.INTRODUCTION

Software process simulation (SPS) modeling is the numerical assessment of a mathematical model that replicates the actionsof the application development process under consideration. SPS can handle the inherent risk and randomness in software development and model its dynamic nature. The software industry has faced numerous allegations of schedule and cost overruns, in addition to poor product quality, from both commercial and government software development groups during the previous few decades. Additionally, the complexity of software and rising customer expectations for "better, faster,cheaper" have significantly "raised the bar" for creators of software to enhance performance. Academic scholars and professionals alike are becoming more interested in software process modeling through simulation as a method of delving into difficult business and policy issues. Although simulation modeling has been used for many years in a variety of areas, it has only lately been used in the processes for software development and evolution.

Software process simulation modelling is currently beginning to be applied to a range of concerns, from assisting process changes to software project management training to the strategic oversight of software development. Applications for software process simulation span a wide range of life cycle stages, from short-term focused stages to longer-term product evolution models with significant organizational implications

The USC System Factory project and the USC ATRIUM Laboratory have been researching many types of process modeling and simulation difficulties over the past ten years.

These efforts concentrate on comprehending the organizational procedures involved in the creation and application of software systems. Numerous of these initiatives have focused on the useof modeling and simulation tools, methodologies, and conceptsto comprehend and predict software process trajectories beforethey actually occur. When examining both "as-is" (simulated models of software processes found in a particular organizational context) and "to-be" (simulated versions of new or revised software procedures planned for use in the same organization setting), this is done. The same tools, though, have also been employed by us to comprehend, validate, and improvemodels of Using empirically supported data gathered fromactual process performances, we analyze software processes. Itis also possible to use similar tools to describe and simulate the"here-to-there" processes and changes that take place as an organization transitions from its current state to that of its futuresoftware processes. This essay illustrates the strategy, the encounters, and the insights discovered during these endeavors.

II.BACKGROUND

The main phrases from the title—"process," "model,"

and "simulation"—as well as what they represent whencombined are covered in this section. A logical framework of people, technology, and practices that are arranged into work activities and intended to transform information, materials, and energy into certain end results(s) is known as an organizational (e.g., business) process (derived from Pall, 1987). Software development and evolution, system engineering design, travel expense reimbursement, task selection and funding, and project management are just a few examples of organizational/business processes.

An individual software creation, maintenance, or evolution process is the focus of a simulation model for software processes. It can depict such a process in its currentstate (as-is) or in its future state (as-to-be). Because any models are abstract concepts, a model only depicts a small portion of all the conceivable components of a software process that may be modeled, specifically those that the model developer felt were particularly pertinent to the problems and concerns the model was intended to answer.

Simulation models frequently serve as a foundation for experiments, anticipate behavior, provide "what if" answers, impart knowledge about the system they are modeling, etc. Typically, these models are quantitative, but this isoccasionally the case.

A digital model that embodies the aforementioned traits and simulates a dynamic system or event is known as a simulation model. When the costs, hazards, or logistics of influencing the actual system of interest are prohibitive, oneof the key reasons for creating a simulation model or utilizing any other modeling technique is that it is a cheap approach to get crucial insights. Typically, simulations are used when a system's complexity is too great for static representations or other methods to effectively depict.

III.VALUE OF PROCESS SIMULATE

The complexity of goods and manufacturing procedures has made "time-to-market" and asset optimization more difficult for top manufacturers. Teams of manufacturing engineers are expected to meet cost, quality, and start-of-production goals while facilitating faultless new product introductions. Leading manufacturers use their organizational expertise and the accessibility of 3D models of their products and resources to digitally evaluate their manufacturing processes in advancein order to address these difficulties.

To assure upfront production optimization, hundreds of validation experiments may be carried out quickly and nearlyautomatically thanks to new, developing technology.

Designing and validating manufacturing procedures in a 3D changing setting is made easier by Process Simulate. Manufacturing engineers can reuse, write, and validate manufacturing processes thanks to Process Simulate's complete integration with the manufacturing backbone. A cutting-edge 3D environment called Process Simulate offers the ability to simulate realistic behavior in manufacturing processes and optimize cycle times and process order. Process Simulate makes it easier to simulate assembly processes, human actions, and the mechanical actions oftools, gadgets, and robots. Process Simulate is extremely scalable, giving multiple engineering disciplines the information and toolkit needed to review intricate processes and confirm them from numerous angles and phases.

3.1 Simulation and modeling software development processes

When formalised as computational descriptions, software process models are very intriguing (Curtis et al., 1992). As a result, we can model, browse them interactively, and execute them symbolically. In layman's words, this means that simulation involves the assigned agent performing process tasks symbolically while using the tools, systems, and resources to create the desired products. The manager's agent, for instance, would "perform" her management tasks in a simulation of a software project management process. The precedence arrangement of the steps or sub-tasks that are specifically specified in the modelled process instance determines how tasks are modelled. Then, as they proceed, agents and tasks may use or absorb simulated time, budgeted money, and other resources. The simulation advances as long as job preconditions or postconditions are satisfied at each stage because tasks along with additional resources can be modelled with arbitrary levels of accuracy and depth. In order for a manager to assign a sta to the task of producing reports, for instance, a sta has to be available at that same moment; otherwise, the simulated process halts, reports the issue, and then

waits for more input or instructions to the simulation user.

Knowledge-based simulation (KBS) and discrete-event simulation (DES) are two types of simulation technologies that we have used in our work at USC to study software development processes (Mi and Scacchi, 1990; Scacchi and Mi, 1997). Beginning in 1988, we developed our own KBS tools while using a DES package that was readily available on the market. Each is explained individually.

Companies like Knowledge Based Systems (www.kbsi.com) and Intelligent System Technology (www.intelsystech.com) now offer commercial KBS products integrated with DES. As previously mentioned (Noll and Scacchi, 1998; Scacchi and Mi, 1997; Scacchi and Noll, 1997), we have also created and used tools that enable users to explore, gradually simulate (traverse), and execute computer process models across dispersed network settings.

3.2 Knowledge-based simulation of software processes

One type of technology for simulating software processes is KBS. The running of a finite ®nite state machines that iteratively navigates explicitly or implicitly modelled states and/or events in the simulated process instance is central to many methods for modelling and simulating complicated processes. The entity process approach created by Kellner, for instance, employs the State chartstm tool to formally describe and traverse the states specified in a software process (Humphrey and Kellner, 1989). Simulations frequently also apply to actual instances of a modelled process, with the model being created first and then being put through the simulation once its parameter values have been set. Software process states may be expressly expressed or implicitly represented in a KBS.

Implicit representation describes how the values that make up a snapshot of the underlying knowledge-base of interconnected objects, attributes, and relations used to record and control the simulation correspond to the process states. A class of process models or a process paradigm without instance values can also be symbolically evaluated by KBS. Last but not least, KBS use pattern-directed inferencing, which is accomplished using a production rule basis (Mi and Scacchi, 1990).

These rules keep track of and update the knowledgebase's objects, attributes, and relations' values. The implicit condition of an operation at any one time is the collection of every value in the knowledge base.When any number of rules are turned on and ®res, the computational action of the rule is carried out, and the knowledge-based process state is updated. In order to start differentiating KBS methods to software process simulation, these features are helpful.

KBS is helpful in addressing various simulated process execution behaviour kinds. We have identified four interesting categories of process behaviour. The first step is to comprehend software processes that demand fine granularity. Second, examining processes with various or mixed granularity levels.

The third step is to examine the ways that software developers (or agents) connect and work together. Analysing processes with constantly changing structure and control is the fourth step (Mi and Scacchi, 1993).

Granularity in this context refers to the level of detail that we attempt to recreate in order to comprehend the dynamics or specifics of the gross or one-level process.

4.1Process Simulate in different segments of manufacturing process

The various steps of the production procedure can be verified using Process Simulate. It is possible to mimic virtual manufacturing zones by simulating assembly processes, human activities, welding, continuous processes like laser welding and gluing, and other robotic procedures within the same environment.

Robotic controls, PLC logic, and realistic human behavior are all modeled in the simulation.

4.1 Process Simulate Assembley

Users have the ability to check the viability of an assembly process using Process Simulate Assembly. Manufacturing engineers can use it to find the quickest cycle time and mosteffective assembly order while accounting for collision clearance. By running virtual reach testing, collision analysis, and simulating the entire assembly process of the good and the tool together, Process Simulate Assembly offers the potential to find the best tool for the

process.

4.2 Process Simulate Human

Users can utilize Process Simulate Human to validate a workstation's design to ensure that the product parts areaccessible for assembly, maintenance, and repair. Process Simulate Human offers strong capabilities for ergonomics analysis and optimization, ensuring an ecologically safe process in accordance with industry requirements. The user can simulate human tasks accurately using the tools for human simulation, and process cycle times can be optimized using ergonomics libraries that are accepted in the industry.

4.3 Process Simulate for Spot Weld

Users of Process Simulate Spots Weld can create and test spotwelding procedures in a 3D visualization and simulations environment, from the first planning stage all the way through to the intricate design stages and offline programming. Process Simulate Spots Weld makes it easier for industrial engineers to complete activities like allocating weld sites to stations that take into account geometry and cycle time restrictions and choosing the optimum weld gun using a classified collection to reuse current guns and tools.

4.4 Process Simulate for Robotics

Users of Process Simulate Robotics can create and simulate extremely intricate robotic industrial environments. The exceedingly complex operation of synchronizing many robot zones is made simpler by Process Simulate technologies like the cyclic event evaluator and simulated customized robot controller. The tools for robotics simulation enable the creation of a path free of collisions for every robot and the optimization of their cycle times.

4.5 Process Simulate for Commissioning

The existing production and engineering data can be streamlined using Process Simulate Commissioning, fromconceptual design all the way to the shop floor. For the mechanical and electrical disciplines involved in actual production zone/cell commissioning, Process Simulate Commissioning provides a uniform integration platform. By simulating real PLC code with real hardware utilizing OPC and real robot programs, users of Process Simulate Commissioning may create the most accurate virtual commissioning environment.

V. WHAT IS THE GOAL OF PROCESS SIMULATION

Increasing process efficiency ultimately results in improved profitability and a competitive advantage, which is the aim ofprocess modeling in corporate operations.

Process simulation is an effective technique for process optimization since it may simulate several situations withoutinterfering with ongoing business as usual. The simulation may be used to repeat and enhance processes and tweak to determine what works best by building a virtual copy of a process. By doing this, firms may spot possible issues and areas for improvement before putting them into practice.

The goal of process simulation is to establish a safe, virtual environment in which each of those attributes can be tested and refined across a full range of production levels

A. Anticipate process behavior

B. Variation of parameters

C. Calculation of process variables

D. Production of a graphical process model.

This things help to achieve the ultimate goal in the reqired sectors for the following variable constants and this increases effectively for various causes, Teams of manufacturing engineers are expected to meet cost, quality, and start-of-production goals while facilitating faultless new product introductions. Leading manufacturers use their organisational expertise and the accessibility of 3D models of their products and resources to digitally evaluate their manufacturing processes in advance in order to address these difficulties.

To assure upfront production optimisation, thousands of validation experiments may be carried out quickly and nearly automatically thanks to new, developing technology.

VI.WHY IS PROCESS SIMULATE IMPORTANT?

Process simulation can be used to create better strategies for achieving process excellence and identifying areas where anorganization's processes

need to be improved.

As part of an analysis project, process modeling may be employed as well to determine potential hazards connected to new initiatives as well as optimize current operations. Businesses can boost profitability by investing the time in carefully analyzing and simulating processes.

VII.PROCESS SIMULATE INVOLVES SIX STEPS

1. Process Mapping - This entails developing a visual depiction of the entire process, including all of its steps, duties, and decision-making opportunities. It is commonpractice to create this depiction using flowcharts or specific BPM software tools.
2. Data Gathering - Gather pertinent information about the simulation process, including task durations, accessibility of resources, and costs. This information is utilized to create baseline performance measures and to provide the simulation model with realistic inputparameters.
3. Model Development - Based on the procedure map and the information acquired, build an equation or computermodel of the process. The linkages among tasks, resources, and process decision-making points should be appropriately reflected by this model.
4. Simulation Runs - Make use of the model for simulatingseveral situations, some of which can entail adjusting thedistribution of resources, job durations, and process flows. This process aids in locating possible roadblocks, inefficiencies, and growth opportunities.
5. Analysis and Optimization - Examine the simulation's outcomes to find trends, patterns, and areas for development. Run further simulations to verify the suggested improvements and modify the process accordingly to improve performance.
6. Implementation and Monitoring - Implement the improved procedure in the actual world and keep an eye on how it performs to make sure the desired results are realized. By allowing for continual adjustments as needed, continuous monitoring fosters a cycle that promotes constant progress.

VIII.HOW PROCESS SIMULATE WORKS

Process simulation software first builds an electronic representation of a business process using computer algorithms as well as data science, and then predicts how the digital representation would respond in various process scenarios.

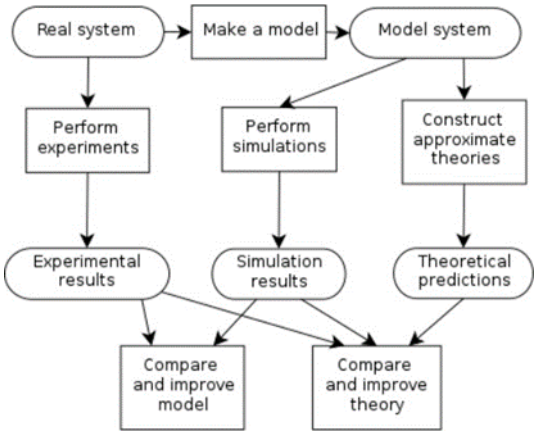

Figure1. Visualisation of a Simulation and Model.

Businesses can find bottlenecks in their operations and make modifications to minimize them by simulating various situations. This may be employed to re-engineer fresh procedures or instruct staff on how to use current systems most effectively.

A digital twin of a company (DTO), which simulates a wholecompany in the actual world, can likewise be made viaprocess simulation.

IX.BENEFITS OF PROCESS SIMULATE

1. Risk reduction: Organisations can test changes virtually before putting them into practise to reduce risks and unforeseen problems.
2. Cost effectiveness: Businesses can save money by finding inefficiencies and testing innovations in a virtual setting before implementing them in actual procedures.
3. Optimal utilisation of resources and enhanced output are made possible through process modelling, which can identify underutilised areas.
4. Flexibility: Businesses may readily test a variety of situations, enabling quick modifications and decision-making agility.
5. Predictive Insights: Simulations can offer helpful knowledge into how processes may react to upcoming difficulties or changes, in addition to existing operations.

X.PROCESS SIMULATION TOOLS

There are numerous well-liked process simulation programs out there that support various sectors and use cases. Some of thesetechnologies are designed specifically for managing business processes, whilst other simulation tools are more general- purpose and can be modified for process simulation. A few well-liked process simulation tools are listed below:

1. Bizagi Modeler - a user-friendly BPM tool with a drag-and-drop interface that makes it easy to create, simulate, and optimize business processes. It supports the BPMN 2.0 standards and offers team collaboration tools.
2. Bonita - a free BPM tool with tools for modeling processes,simulation, and automation. It offers a single development platform for creating and deploying unique applications and supports BPMN 2.0.
3. Simul8 - a general-purpose process simulation tool that may be applied to a variety of industries, including manufacturing, medical care, and logistics. It has a drag- and-drop interface, objects that may be customized, and integrated reporting capabilities.
4. Anylogic - A flexible option for modelling complicated business processes, multi-method simulation software supports continuous event, dynamics of systems, and agent-based modeling. A wide range of industries and applications are catered to by the libraries that AnyLogic provides.
5. ExtendSim - a general-purpose simulation program that supports continuous, discrete rate, and discrete event modeling. Numerous sectors, including manufacturing, medical care, logistics, and others use it extensively.
6. ProcessMaker - a BPM platform that is open-source and offers tools for modeling processes, simulation, and automation. It provides a cloud-based solution for simpler deployment and collaboration, and it supports BPMN 2.0.

XI.PROCESS MODELLING

Modelling discrete processes based on events (such as actions and activities) is the focus of process modelling. According to Guizzardi and Wagner (2013), behaviours and operations are particular kinds of occurrences while objects and events are the ontologically essential categories.

Gantt charts, which enable the modelling of actors and the sequences of their activities, were the origin of process modelling.

Conditional branching ("decisions") and activity sequences are combined in flowchart languages. The parallel branching, actors (sometimes known as "swim lanes"), and constrained actors in UML Activity Diagrams (ADs) extend flowcharts. While BPMN expands on ADs by providing a more generic concept of events (including asynchronous messaging events and event-based branching), ADs only support the concept of events.

11.1 MODEL-DRIVEN ENGINEERING

Model-driven engineering (MDE) is a universal strategy that can be used in all engineering disciplines. It is founded on the idea that the main engineering artefacts for collecting concepts, logical designs, and carrying out designs for technological products are models.

As a result of the Unified Modelling Language's (UML) success in IS/SE, UML-based MDE has gained traction in these domains, where information modelling is the primary modeing issue (Whittle et al. 2014). However, in order to acquire full models as a foundation for code generation in simulation engineering, much as in process engineering, information models must be supplemented by process models.

Although the use of process modelling in MDE is still a relatively new practise, MDE using UML class modelling is already a well-established engineering practise in the software industry. research topic that has primarily been looked into in workflow control and service engineering fields. MDE has not received much attention in simulation engineering to date for a number of reasons, notably a lack of set foundations and standards.

11.2 Model-Driven Software Engineering

In MDE, there is a clear distinction between three kinds of models as engineering artifacts resulting from corresponding modeling activities in the analysis, design and implementation phases:

1. solution-independent domain models (also called conceptual models),
2. platform-independent design models,

3. platform-specific implementation models.

In the analysis stage of a software development project, domain modelling are solution-independent representations of a problem domain or of a system under examination. In conceptual data models and conceptual process models, a domain model may comprise both explanations of the domain's state structure and its operations. They are "computation-independent," or independent of solutions, in a way that they are not bothered by system design decisions or other computational concerns. Instead, they concentrate on the viewpoint and vocabulary of the subject area specialists for that field in question. On the foundation of the domain model, a platform-independent development model is created throughout the design phase as an overall computational solution (for an application platform or for a simulated system).

It is possible to create a variety of design models using the same domain model.

Then, a number of platform-specific implementation frameworks are derived from the design model by selecting one or more target platforms for technology and taking into account implementation issues like styles of architecture and ineffective quality criteria to be maximised (e.g., performance, adaptability). These representations can be converted into code right away.

It is possible to collect several technological platforms that adhere to shared concepts, such as object-oriented programming (OO), and regard them as constituting an infrastructure class, for which any "class of platforms" OO model, for example, can be created as a step between a design model and the models for implementation based on it.

These one-to-many relationships between a conceptual model and corresponding design models, as well as between a design model and corresponding implementation models, are illustrated in Figure 1.

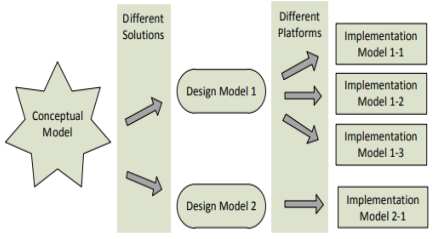

Figure 1: From a conceptual model via design models to implementation models.

In the implementation step, a model for implementation is manually or automatically converted from model to code using the target platform's programming language. The deployed solution is subsequently implemented in a target context after testing and debugging.

A conceptual framework for a (software) structure, often known as a "(software) system model," does not include a single model diagram that represents all perspectives or facets of the system that needs to be produced (or defined). Instead, it consists of a collection of designs, at least one for every various point of view. The three modelling stages of conceptualization, design, and implementation are all interconnected by the two most crucial points of view:

1. information modelling, which is concerned on the state framework of the domain or system being looked into (SUI);

2.Process modelling, which focuses on the domain's or SUI's dynamics

Entity Relationship (ER) Diagrams and UML Class Diagrams are two popular information modelling languages. UML class diagrams can be used to create any types of information models, including SQL database models, because the latter encompass the former. Examples of frequently employed Petri Nets, UML State Charts, UML Activity Diagrams, and the UML Modelling Language are used as BPMN stands for business process modelling notation.

The fact that there are more alternative process modelling languages shows that there is more consensus on the correct principles for information modelling than for process modelling. This seems to suggest that there is less comprehension of the nature of things and procedures than understanding the relationships between objects. Some modelling languages, such as BPMN and UML Class Diagrams, have customizable variations that can be applied to all three modelling levels. Other languages were created to only be utilised on one or two of these levels. Petri Nets, for instance, cannot be used to simulate conceptual processes.

XII.CONCLUSION

Overall, the type of processes used can be used to describe our encounters with these modeling capabilities. In order to facilitate exploratory study of software process models, we discovered that knowledge-based simulations was most beneficial.

For most individuals, knowledge-based virtual reality may be "rocket science." But when utilized to comprehend fine-grained or deep causal relationships of infrequent or loosely structured software operations, it can be helpful. This can make it easier to gain qualitative insights intohow software processes work. Contrarily, utilizing coarse-grained and statistically representative data samples, discrete- event simulation was most useful for verifying and evaluating shallow process models with instances of high rate, short cycle- time processes. This makes it easier to gain quantitative insightsinto how different software process instances behave. As a result, we discover that both When used to enhance one another's strengths, knowledge-based or conventional discrete-event simulation skills are beneficial.

REFERENCE

[1] Akhavi, M., Wilson, W., 1993. Dynamic simulation of softwareprocess. In: Proceedings of the 5th Software Engineering Process Group National Meeting, Costa Mesa, California, April 26±29. Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA.

[2] Curtis, B., Kellner, M., Over, J., 1992. Process modeling.Communications of the ACM 35 (9), 75-90.

[3] Gruhn, V., 1992. Software process simulation on arbitrary levelsof abstraction. Computational Systems Analysis, pp. 439-444

[4] Gruhn, V., 1993. Software process simulation in MELMAC.SAMS 13, 37±57.

[5] Hansen, G., 1997. Automating business process reengineering: Using the power of visual simulation strategies to improve performance and pro®t, 2nd ed. Prentice-Hall, Englewood Cli€s,NJ.

[6] Heineman, G., 1993. Automatic translation of process modeling formalisms, Technical Report CUCS-036-93. Department of Computer Science, Columbia University, NewYork, NY.

[7] Kellner, M., 1989. Software process modeling: value and experience. In: SEI Technical Review. Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, pp. 23-54.

[8] Kellner, M., 1991. Software process modeling support for management planning and control. In: Proceedings of the First International Conference on the Software Process, Redondo Beach, California, October 21-22. IEEE Computer Society Press,Los Alamitos, CA, pp. 8-28.Chan, Cawin. “Model Compression: A Look Into Reducing Model Size.” Medium, 14 Oct. 2021.

[9] Pall, G., 1987. Quality process management. Prentice-Hall,Englewood Clis, NJ.

[10] Paulk, M., et al., 1993. Key practices of the capability maturity model, Version 1.1, Technical Report CMU/SEI-93-TR-25. Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA.

[11] Mi, P., Scacchi., W., 1992. Process integration in CASE environments. IEEE Software 9 (2), 45±53. Reprinted in: Chikofski, E. (Ed.), Computer-aided Software Engineering, 2nd ed. IEEE Computer Society, 1993.

[12] Garg, P.K., Mi, P., Pham, T., Scacchi, W., Thunquest. G., 1994. The SMART approach to software process engineering. Proceedings of the 16th International Conference on Software Engineering, Sorrento, Italy, pp. 341±350.

[13] Mi, P., Scacchi, W., 1996. A meta-model for formulating knowledgebased models of software development. Decision Support Systems 17 (3), 313±330.

[14] Noll J., Scacchi., W., 1998. Supporting software development in virtual enterprises. Journal of DigitalInformation (http://journals.ecs.soton.ac.uk/jodi/) 1 (4).

[15] Scacchi, W., Noll, J., 1997. Process-driven intranets: Life-cycle support for process reengineering. IEEE Internet Computing 1 (5), 42±49.

[16] Scacchi, W., Mi, P., 1997. Process life cycle engineering: A knowledgebased approach and environment. Intelligent Systems in Accounting, Finance, and Management 6 (1), 83±107