A Software Engineering Approach and Tool Set for Developing Internet Applications

David A. Marca and Beth A. Perdue
OpenProcess, Inc.
127 Rand Terrace
Auburndale, MA 02466 U.S.A.
01-617-641-9474
dmarca@openprocess.com

ABSTRACT
If a business built a plant to produce products without first designing a process to manufacture them, the risk would be lack of capacity without significant plant redesign. Similarly, lacking a software engineering approach and tools for designing e-business connections before creating them, can risk: 1) designing the business partnership incorrectly, 2) not implementing the connection quickly enough, or 3) having operations that cannot adapt to changes in business direction. This paper presents a software engineering tool for developing process-oriented Internet applications that implement e-business connections. It gives an approach for using this tool in conjunction with standard commercial IDEF0 tools to create adaptable connections. It is organized to match a formal demonstration that shows the step-by-step usage of these tools, and cites software engineering principles that, when applied, ensure adaptability.

Keywords
Software Engineering Environment, IDEF0 Process Model, Internet Application, Client/Server Architecture, Hypertext

1 SOFTWARE ENGINEERING FOR E-BUSINESS

E-business connections can be defined by assessing their effect on a supply chain – the network of suppliers and customers within which a business operates [1]. For example, a computer manufacturer has a chain of suppliers for disk drives, monitors, software, etc and a chain of customers in its retailers, resellers and users. E-business connections can implement any portion of a supply chain. For simplicity, they can be defined as market-focused or production-focused [2]. Market-focused connections have a direct and immediate effect on the marketplace.

ICSE 2000, Limerick, Ireland

A production-focused connection is one that outsources a portion of the value of a business, as perceived by the customers. Since others are performing the activities that create the value, the effect of the connection on the marketplace is less direct, and there are time delays. Production-focused connections also require monitoring partner performance [3]. With the right design, an e-business connection can establish a standard for partner performance and for performance monitoring. A production-focused connection is the running example.

The question for software engineers is: "How can business factors and rules of engagement for an e-business connection be managed at all levels by both partners?" The answer is a software engineering approach and tool set that produce an *architecture* of the business relationship, adaptable processes within that architecture [4], and both supported by adaptable technical infrastructure [5]. "Adaptable" means that all designs constantly mirror and support the business partnership. Internet technologies have made this kind of infrastructure (i.e. supply chain management through electronic commerce) a possibility [6]. Research and our experience indicate that adaptable e-business connections can arise when the following are created *in order*:

- <u>Business Architecture</u> for the partnership.
- <u>Business Design</u> that defines all essential policies.
- <u>Process Architecture</u> designed for adaptability.
- <u>Process Design</u> for the activities of the connection.
- <u>Technical Architecture</u> that enables adaptability.
- <u>Technical Design</u> that supports process activities.

Due to their ability to distinguish rules from data flows, IDEF0 tools [7] are used for business architecture and design. The OpenProcess™ tool is used to translate IDEF0 models into processes that are viewable and "executable" via an Internet browser. This tool also links the resulting Internet-based processes to software applications whose modularity is a 4-tier logical / 3-tier physical client/server architecture [8]. The rest of this paper will detail these software engineering tools in the context of how they are used in each of the six aforementioned steps.

2 BUSINESS ARCHITECTURE

Business Architecture defines: the boundaries among major business activities, the fundamental business policies, and the external market forces. A boundary describes the roles of partners in a supply chain and the activities each must perform. The policies establish the cycle time for the activity. External factors are external events that affect the cycle time. Customers, suppliers, substitute products, rivalry, new competition, regulatory shifts, economic cycles, etc can appear in the architecture [9].

The value of modularity and well-defined sub-system interfaces is widely known [10]. These software engineering principles have great applicability when developing a business architecture. For example, Figure 1 is a business architecture for a product-oriented company of five departments. Let us assume that executives choose to outsource "Product Manufacturing and Distribution." Box three in the IDEF0 diagram of Figure 1 therefore defines how both companies are expected to accomplish this collaboration [11] via the business-to-business interface.

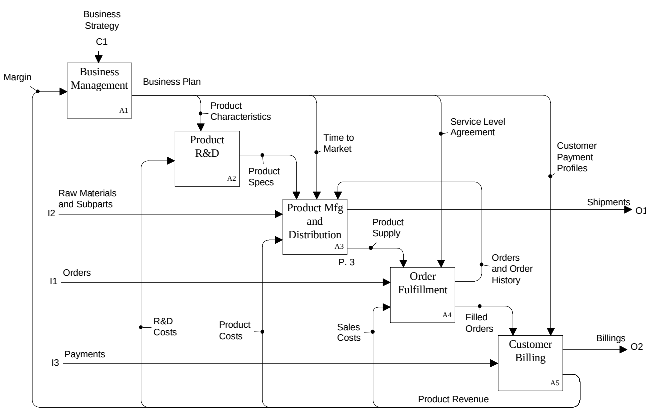

Figure 1: A Product-Oriented Business Architecture

3 BUSINESS DESIGN

Business Design defines how business units respond to business events. This design is created *within* the context of a business architecture. Thus, events and their responses are already identified as the parameters [12] of the business-to-business interface. The architecture also provides the design with rules that govern how responses are met. These rules include service level agreements, defining response times. This way, executives create business policies that are carried out in a time frame that ensures competitiveness.

Figure 2 is a business design for the supplier of "Product Manufacturing and Distribution." Service level agreement (SLA) "Time to Market" defines how long before manufacturing begins. The design shows the supplier receiving "Sales Volumes" for adjusting manufacturing capacity, and "Sales Trends" for adjusting inventory levels. Since the implementation can deliver such process control data in real-time or near real-time speeds, this helps the supplier respond faster to changing market conditions [3].

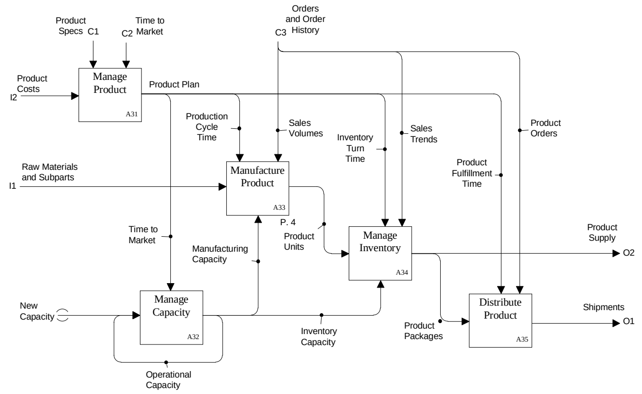

Figure 2: A Business Design with Service Levels

4 PROCESS ARCHITECTURE

So far, we defined business units and activities, and made decisions as to where e-business connections will occur and which policies define expected performance. The connection can still fail, if either company cannot adapt its operations quickly and easily to respond to changes in business events, business cycles or market conditions – even if those changes occur elsewhere in the supply chain [13]. Adaptability is achieved when an operation has distinct steps, each fully capable of responding to a single business event.

One process architecture [14] is created for each box in the business design. Each is an IDEF0 model, whose decomposition isolates those operational steps that fully respond to a single, low-level event. Each architecture is adaptable because: 1) The loss of a business event means the loss of only a single operating step. 2) When a service level agreement changes, the steps governed by that agreement are easily identified for alteration [15].

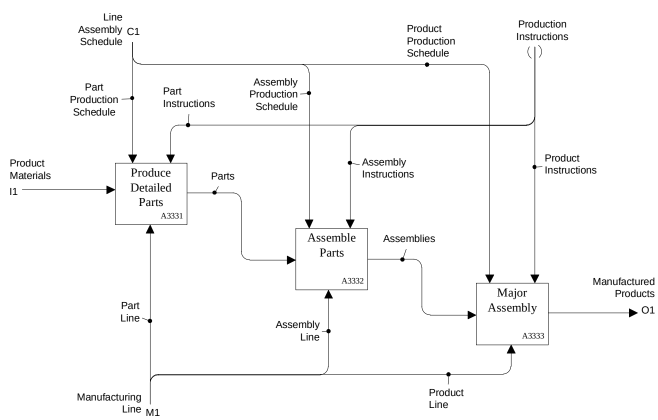

Figure 3. Process Architecture for Low Level Events

Figure 3 is an architecture for Produce Detailed Parts. Its event is Product Materials and its response is Parts. Part

Instructions control how parts are made. Part Production Schedule, the performance metric, is tied directly to the Time to Market SLA via arrow decompositions and transformations in the IDEF0 models. At this point, IDEF0 tool usage stops, and OpenProcess™ tool usage now begins.

5 PROCESS DESIGN

Partnering companies can fail to anticipate conflicts or lack of accountability. When e-business connections lack clearly defined process, outcomes become counterproductive. This can range from gridlock or stalemate, to an unimaginative compromise where no one is really committed. Worse, one side may impose its will on the other. To avoid such outcomes, a process design model is built by defining clear policies, constraints, response times, etc, for each process step. Traditionally, such models are built to work *with* hypercode [16], but our tool creates this model *as* hypercode.

To accomplish this, IDEF0 models are first loaded into our tool. This creates a hierarchy – the top being the business units, the bottom being process steps. A description template [7] is then created, so it can be applied to each step during the load. It defines eight different aspects of process, thus organizing descriptions into small, easy-to-understand units. Some aspects of product-oriented connections are oplicy, exceptions, roles, ISO9000, controls and metrics.

Figure 4 shows our tool being used after an IDEF0 model load. It identifies an internal metric [17] "Parts Per Day" for "Produce Detailed Parts." Part production partially indicates 1) if the "Time to Market" SLA is being met, and 2) if production capacity is not being exceeded. Once descriptions are complete, our tool generates a set of web pages.

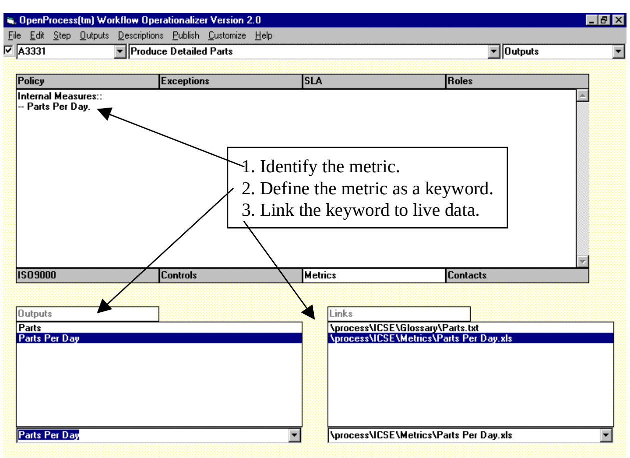

Figure 4: Defining Process for One Operational Step

6 TECHNICAL ARCHITECTURE

Software engineering principles for adaptable client/server architecture are well-known [5]: 1) Design business units so they do not cross operational boundaries. 2) Place the user interface on the client machine. 3) Allocate one server for the operating rules for each business unit. 4) Create one stored procedure for each operating rule. 5) Isolate the data access logic from stored procedures. 6) Allocate a server or the mainframe for the data access logic and the database.

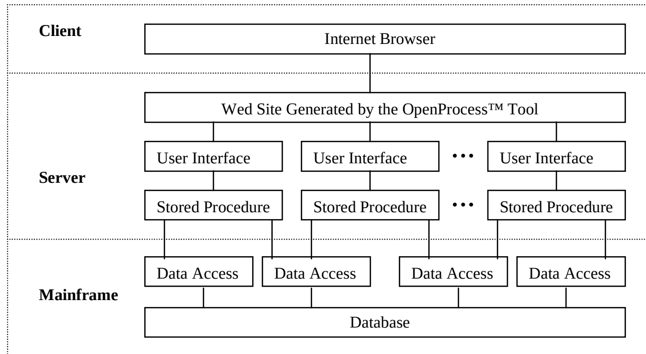

Figure 5: Technical Architecture for OpenProcess™

Our tool refines this approach by requiring: 1) Many small user interfaces be created instead of a single, large user interface – one small user interface for just one process step. 2) The web pages created by our tool be placed on the server for that business unit. 3) Our tool be used to link process keywords to the small user interfaces or to files of live process data. Figure 5 depicts the refined architecture.

7 TECHNICAL DESIGN

The last step in our approach is to design the many small user interfaces and the applications that generate live process data, and to then use our tool to link process keywords to those user interfaces or data files. Once this is done, people "run" the process by simply accessing the web site created by our tool via their Internet browser. They then simply click on a highlighted keyword that is a hyperlink to the application, repository artifact, or live data that is relevant to the process step they are currently viewing [18].

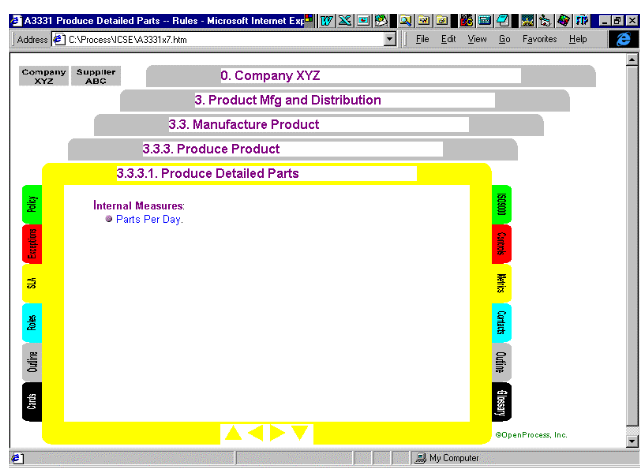

Figure 6: Internet-Based Process Viewed as "Cards"

Figure 6 shows how "Produce Detailed Parts" would appear in an Internet browser after our tool creates its web pages. Notice how the IDEF0 model has been translated into a set of "cards" – one card for each process step. Also notice how each card has "tabs" – one tab for each process aspect chosen in the Process Design step of our approach. Lastly, notice the "Parts Per Day" metric is encoded as a hyperlink to permit "drill-down" [19].

When users click on "Parts Per Day," they accesses the live data to which this link refers. In this case, it is a graph (Figure 7). The click activates a new Browser in which the user examines production data over time (e.g. when production fails to meet the service level agreement or when it exceeded manufacturing capacity). This way, users only see the application user interface, repository artifact, or live data relevant to the process step they are currently viewing.

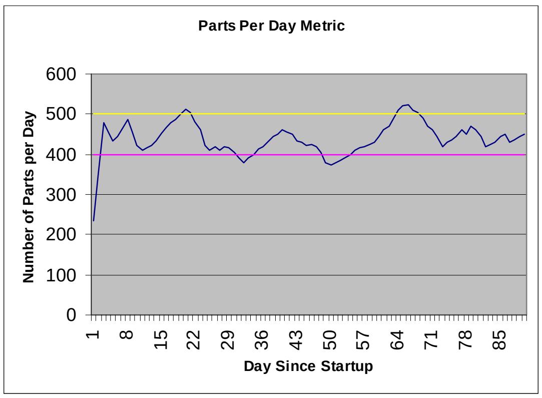

Figure 7: A Hyperlink Accesses Live Process Data

8 HYPERLINK SEMANTIC KNOWLEDGE

Hyperlink semantics are very difficult to design [20], implement [21] and use [22]. Our tool solves this problem by creating web pages from a rigorous process model, having four arrow types: 1) input, 2) control, 3) output and 4) mechanism. The tool creates a hyperlink for each arrow, and lets the software engineer assign a color to each arrow *type*. This results in web pages whose multi-colored links give users knowledge as to whether they are clicking on: 1) data used by a process step, 2) a business rule, 3) data generated by a process step, or 4) a work instruction [23].

Internet browser customizations can be set to alter hyperlink appearance. Novice users (e.g. new workers) keep the multi-colored links as they learn the process. Infrequent users (e.g. managers) override the multi-color scheme to see all links in one color – thus creating reminders. Expert users (e.g. supervisors) turn off link colors altogether and activate the hover color. Then, process descriptions read like plain text, yet the mouse "lights up" a keyword when it touches it. This combination of IDEF0, OpenProcess™ and Internet browser tools is a dynamic solution to portraying link semantic knowledge *consistently* across novice, intermediate and expert user populations. Such adaptable viewing is rarely supported by Internet-based applications [21] or by process technology such as workflow [11].

REFERENCES

1. Gomes-Cassares, Benjamin, The Alliance Revolution, Harvard University Press; 1998
2. Moore, G., Crossing the Chasm, Harper Business, 1999
3. Poirier, Charles C., Advanced Supply Chain Management, Publishers' Group West, 1999
4. Marca, D., Perdue, B., "Business-to-Business Connection" Soft Eng Tech Council, Vol 15, No 3; 1997
5. Gold-Bernstein, B., Marca, D. Designing Client/Server Systems Prentice-Hall; 1997
6. Kalakota, R., Whinston, A., Frontiers of Electronic Commerce, Addison-Wesley; 1996
7. Marca, D., McGowan, C., IDEF0: Business Process and Enterprise Modeling, Eclectic Solutions; 1992
8. Smythe, C. Internetworking: Designing the Right Architectures, Addison-Wesley; 1995
9. Jayachandra, Y., Re-Engineering the Networked Enterprise McGraw-Hill; 1994
10. Britton, K., et al, "A Procedure for Designing Abstract Interfaces for Device Interface Modules" 5th Int Conf On Soft Eng; Mar 1981
11. Marca, D. "Requirements Elicitation and Specification for CSCW" in Software Engineering Productivity Handbook, Keyes J., McGraw-Hill; 1992
12. Beck, K., et al, "Industrial Experience with Design Patterns" 18th Int Conf On Soft Eng; Mar 1996
13. Taylor, David A.,Business Engineering with Object Technology, John Wiley & Sons, Inc.; 1995
14. Hitomi, A., et al, "A Process System Infrastructure" 19th Int Conf On Soft Eng; May 1998
15. Leonhardt, U., et al, "Decentralised Process Enactment in a Multi-Perspective Development Env" 17th Int Conf On Soft Eng; Apr 1995
16. Kaiser, G., et al, "An Architecture for WWW-based Hypercode Envs" 19th Int Conf On Soft Eng; 1997
17. Randall, R. Randall's Practical Guide to ISO9000 Addison-Wesley; 1995
18. Ackerman, M., Halverson C., "Reexamining Organizational Memory" CACM, Vol 43, No 1; 2000
19. Jern, M., "Information Drill-Down Using Web Tools" in Virtual Worlds on the Internet by Vince, J., Earnshaw, R., IEEE Computer Society Press; 1998
20. Jourdan, M., et al, "Authoring Techniques for Temporal Scenarios of Multimedia Documents" in Handbook of Internet and Multimedia; IEEE Press; 1999
21. Wynblatt, M., et al, "Multimedia Applications on the Internet" in Hndbk of Internet and Multimedia; 1999
22. Trigg, R., "Tools for Communicating in a Hypertext Environment" the Conf on CSCW; Sep 1988
23. Jackson, M., Zave, P., "Deriving Specs from Reqts – an Example" 17th Int Conf On Soft Eng; Apr 1995