

COMPOSING APPLICATIONS WITH THE EMC DOCUMENTUM xCELERATED COMPOSITION PLATFORM

A Detailed Review

Abstract

This white paper introduces the principles, conceptual approach, and benefits of composing applications with the EMC[®] Documentum[®] xCelerated Composition Platform (xCP). It discusses common design patterns and illustrates the approach with a concrete example in the area of Case Management.

January 2012

Copyright © 2009, 2012 EMC Corporation. All Rights Reserved.

EMC believes the information in this publication is accurate of its publication date. The information is subject to change without notice.

The information in this publication is provided “as is”. EMC Corporation makes no representations or warranties of any kind with respect to the information in this publication, and specifically disclaims implied warranties of merchantability or fitness for a particular purpose.

Use, copying, and distribution of any EMC software described in this publication requires an applicable software license.

For the most up-to-date listing of EMC product names, see EMC Corporation Trademarks on EMC.com.

All other trademarks used herein are the property of their respective owners.

Part Number h4723

Table of Contents

- Executive summary.....4**
- Introduction4**
 - Audience..... 4
- What is the Documentum xCelerated Composition Platform?5**
 - Benefits..... 6
- Goals and principles.....6**
 - Goals..... 6
 - Design principles..... 7
- Conceptual approach8**
- Design patterns..... 10**
 - Process design patterns 10
 - Process initiation 10
 - Process data mapping..... 11
 - Process invocation 12
 - User interface patterns 13
 - Task View 13
 - Configurable actions 15
 - Reporting patterns..... 15
 - BAM Control Process 16
 - Data source builder 16
 - Dashboard builder 18
- Case Management example 19**
 - Personas 19
 - Application narrative 19
 - Reusable components..... 19
 - Processes..... 20
 - Anatomy of a process 21
 - The “four stroke engine” 24
- Conclusion 24**
 - Cost reduction..... 24
 - Risk avoidance 25
 - Faster time to market..... 25
 - Ease of extensibility 25
- About EMC 26**
 - Take the next step 26

Executive summary

Over the last few years, the technologies of Enterprise Content Management (ECM) and Business Process Management (BPM) have been increasingly integrated and brought to bear on business problems in a coordinated manner. A business process creates and consumes documents and manages the flow of information throughout its lifecycle. However, there are important business problems that necessitate the coordination of multiple business processes as well as multiple documents and data. All these elements must be managed as a single structural unit. Creating such applications poses considerable technical challenges but also promises to deliver important benefits. A prime example is Case Management, in which the case utilizes multiple processes, documents, and data to accomplish its goals. In this white paper we describe a platform that simplifies and accelerates the development of such applications.

Introduction

Traditionally, developing robust line-of-business applications meant a significant investment in custom coding. This required hiring expensive programming talent, and was labor- and time-intensive. In addition, making changes to these applications was fragile and risky. The solutions that resulted were usually one of a kind, not easily reused between customers.

This white paper proposes an alternative to custom coding, namely an integrated application composition platform. The platform provides an operating environment and tooling; pre-built, configurable components; and design patterns for assembling the components into applications. In addition the platform includes a reference architecture, best practices, educational materials, and collateral software. The goal is to simplify application development by replacing custom coding with configuration.

The remainder of this white paper describes the EMC implementation of this approach: the EMC® Documentum® xCelerated Composition Platform (xCP). Documentum xCP delivers a new standard in application development on a single composition platform that combines a fully integrated set of technologies with development and deployment tools, best practices, and a design emphasis on configuration versus coding. The platform enables partners and customers to rapidly build and deploy case-based business applications and solutions with fewer resources, at a lower total cost of ownership (TCO).

Audience

This white paper is intended for CIOs, CTOs, IT professionals, business analysts, and developers. It may also be of interest to anyone on the business side who wants a better understanding of how case-based business applications and solutions can reduce costs, boost efficiency and productivity, and improve the customer experience.

What is the Documentum xCelerated Composition Platform?

Documentum xCelerated Composition Platform (xCP) is a rapid application development platform that enables partners and customers to build dynamic case-based applications and solutions more quickly, with fewer resources, and at a substantially lower cost. It delivers agile, flexible applications that reduce TCO. Figure 1 illustrates the components of the Documentum xCP platform.

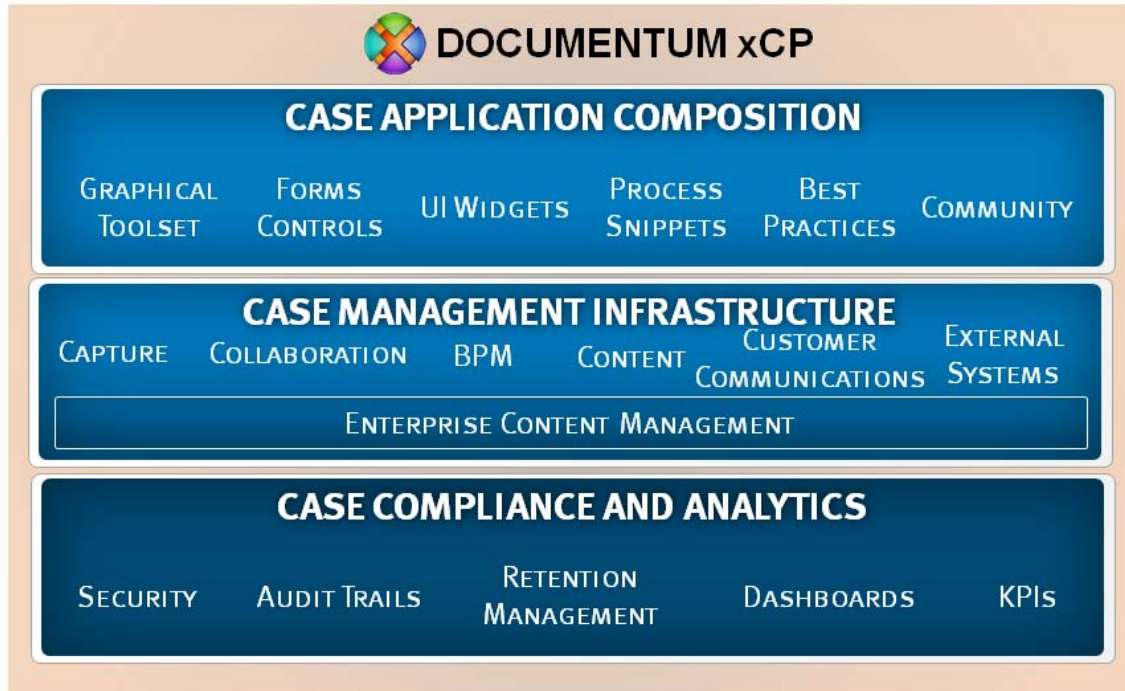


Figure 1. Documentum xCP overview

Documentum xCP provides:

- Integrated tooling, operating environment, repository, and supporting services for content management, case management, compliance, and analytics
- A set of pre-built and easily configured components
- Documented design patterns and composition tools for assembling the components into applications
- Documentation and software “xCelerators” that incorporate best practices to accelerate Case Management and BPM solution development

A key benefit of xCP is to minimize or even eliminate the need for custom code. In effect, coding is replaced by configuration.

Using service-oriented architecture (SOA) principles, xCP creates and manages structured and ad hoc processes, documents, and data, and enables integration with internal and external services.

Benefits

Using the xCP platform will deliver many important benefits to businesses and public sector organizations. Among these are the following:

- Reduce development cost, by employing fewer resources
- Minimize risk, by leveraging configuration and proven design patterns
- Accelerate time to market, because less time will be spent in traditional software development and testing cycles
- Enable solution adaptability and extensibility, through a modular and component-based methodology

Goals and principles

Goals

xCP is designed to create applications that improve the productivity of knowledge workers, the agility of business systems, and the quality of decisions. It does so by supporting these goals:

- Allow participants in a business application to add value based on their expertise and to exercise judgment
- Support these decisions with rigorous and predictable automated processes
- Provide a central point of visibility and control, for tracking, monitoring, and auditing all aspects of the application

The first goal departs from traditional deterministic workflows and allows for ad hoc collaboration and decisions. As pointed out by Gartner analyst Marc Kerremans,

“Case management departs from the traditional view of structured and sequential predefined processes. Instead, workflows are nondeterministic, meaning they have one or more points where different continuations are possible. They are driven more by human decision making and content status than other factors.”¹

The first goal is to provide a flexible, collaborative working environment for the human participants, so that they can exercise their judgment in both an ad hoc manner as well as a more structured, systematic manner.

The second goal reflects the importance of “utility processes.” These automated processes carry out routine functions in the background, eliminating the need for tedious manual functions that do not require human judgment. More importantly, these processes ensure compliance with regulations as well as guarantee the consistency and accuracy of information management.

¹ Marc Kerremans, “Case Management is a Challenging BPMS Use Case,” December 2008.

The third goal enables the management of the application. It requires insight into what is happening right now and visibility into milestones that have been met, and provides metrics on how performers are executing on their responsibilities. Without this level of management and control, such applications would become cumbersome, slow, and unpredictable.

Design principles

Where goals identify what one wants to accomplish, principles focus on how to achieve the goals. xCP is based on three basic design principles.

1. *Provide a rich set of pre-built components and templates that can be extended and used as the building blocks of an application.*

Example 1: xCP includes a wealth of activity templates. Each activity template represents a service or action automatically carried out by the system. By combining these templates into a process, the process designer creates higher level composite business services. The designer drags and drops the template onto the process diagram and then configures the properties of the template. An example is the activity template that creates a new case directly from a folder template.

Example 2: Templates are provided for creating the application's user interface. xCP's TaskSpace user interface is composed of forms, which are based on forms templates. This ensures rapid development and agility: A UI design can be modified in seconds, by dragging and dropping forms controls and setting layout properties. For example, to switch a last name field and a first name field the designer goes into the underlying form, drags and drops, and the change is populated instantly.

Example 3: In xCP's Business Activity Monitor, monitoring reports can be created by dragging and dropping process and data entities, without any need for programming or database knowledge. Creating a bar chart summarizing performance data by a State or Business Unit can be done in less than five minutes. Reports are then dragged and dropped onto dashboards in a manner of seconds.

2. *Create applications through a process of composition; that is, by configuring and then assembling components rather than by custom programming.*

All functions and processes are exposed as services, enabling easy composition into an application. This means that the customer or partner can choose to create a custom user interface and still execute an xCP business process. Since everything is a service, you can invoke these services from the client of your choice.

3. *Define explicit design patterns for assembling components into solutions.*

The classic definition of patterns is due to Christopher Alexander:

“Each pattern describes a problem which occurs over and over again in our environment, in such a way that you can use this solution a million times over, without ever doing it the same way twice.”²

The design patterns define the techniques for combining the components into applications. The section “Design patterns” on page 10 describes several important xCP design patterns. [An extensive set of such patterns](#) is available on the [Documentum Developer Community](#).

Conceptual approach

This section gives a high-level overview of the main ideas and conceptual approach behind xCP and sets the stage for the discussion of the Grants Management sample application.

The core idea underlying xCP is that complex solutions are composed from the interaction of Documentum objects with business processes. The objects provide a persistent, static way of capturing information. The processes are transitory but are a dynamic way of creating, transforming, and moving information. Documentum objects and processes exchange process data. At a high level the interaction between objects, processes, and data can be visualized as shown in Figure 2.

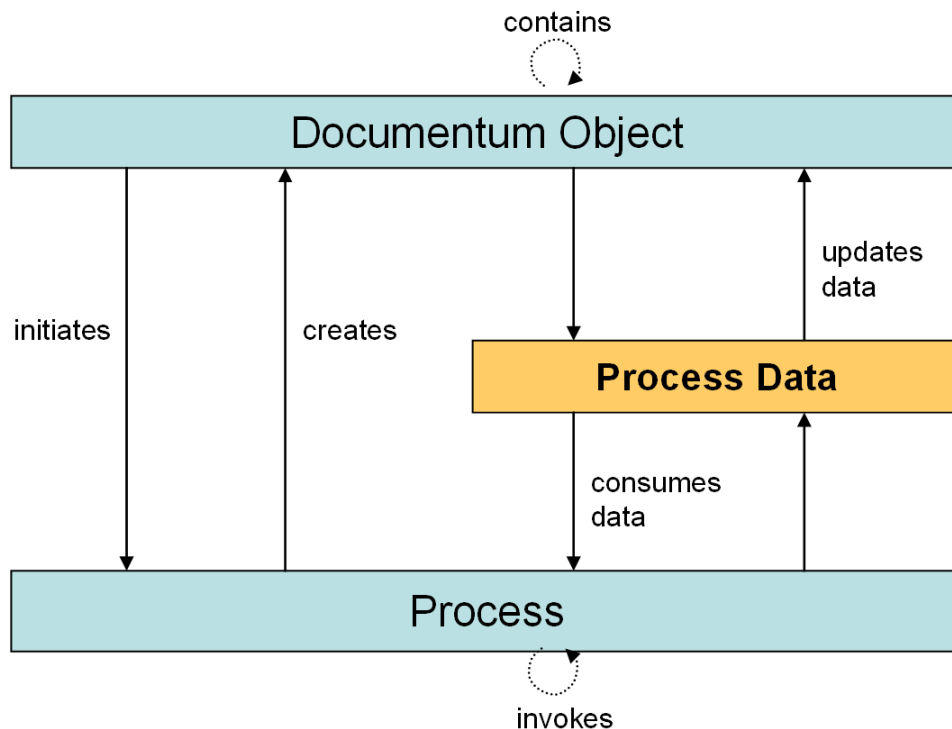


Figure 2. Interaction between objects, processes, and data

² *Design Patterns*, by Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides, 1994

Documentum objects include things such as folders, forms, and documents. Each object contains persistent metadata. Objects can contain or be linked to other objects, such as a folder containing a document.

A process consists of a series of activities, which can be purely automated or present an interface for a human task performer. An activity can manage objects or invoke process instances. Process Data is data that is managed by the process in the form of process variables, which can be of simple or complex type. Process variables are maintained and used over the lifetime of the process and expire when the process is complete.

Events that involve Documentum objects can initiate processes. This can be configured to work in a wide variety of ways.

- Example 1: When a fax is received and is inserted into a folder, a new process instance will be started.
- Example 2: When an e-mail is received, a new process instance will be started.
- Example 3: When data is inserted into a database, a new process instance will be started.

A process consists of a series of activities, each of which can be manual (performed by a person) or purely automated. Activities can create and manage new Documentum objects or invoke instances of other processes.

When users perform an activity in a process, the user interface, TaskSpace, presents data in forms, which the user can read and update. Forms may also contain action buttons, which can be configured to perform various actions. One important type of action is to trigger a business process to carry out a composite business service or automate a human workflow. The task user may be presented with a choice of action buttons. Depending on the choice made by the user, different processes are triggered.

Figure 3 elaborates on the previous figure, highlighting the role of forms in mediating the interaction between objects, processes, and data.

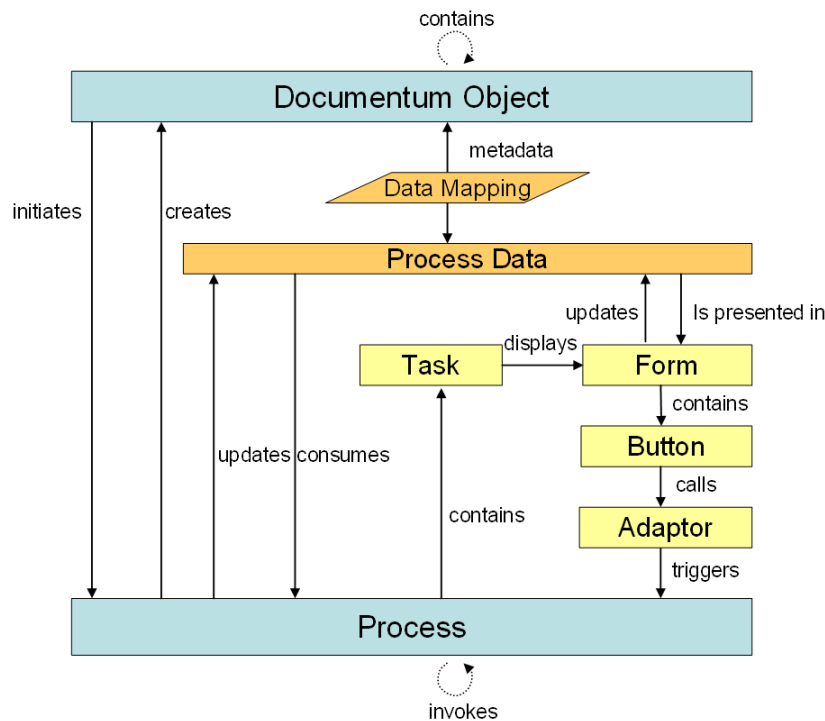


Figure 3. Forms interaction with objects, processes, and data

Design patterns

This section discusses some of the important design patterns for xCP applications. In effect, these patterns specify how to solve recurrent problems using the standard components and templates. Three types of patterns are discussed: process design patterns, user interface patterns, and reporting patterns (needed for Business Activity Monitoring).

Process design patterns

Process initiation

A process instance is started by means of an initiate activity, which can be either manual or automatic. In a manual initiation, the user starts the process instance by submitting a form. The form will contain data fields and possibly attached objects. In an automated initiation, the process is triggered by an event, such as receipt of an incoming e-mail. A listener detects the event and starts the new process instance.

A process can have multiple initiate activities, any of which will start a process instance. As we shall see in the case covering Grants Management on page 19, the Grants Request process can be initiated in any of the following ways:

- HTTP POST
- Receipt of an e-mail
- Incoming FTP

- Web Service request

For automatic initiate activities, the process designer will use the process data mapping tool (see below) to specify the rules for copying data from incoming messages to process data³. For manual initiate activities the process designer will create a form template, which is bound to process data. End users start process instances by submitting this form.

Process data mapping

Processes enable the exchange and integration of information between diverse systems and data sources.

- Example 1: A process may make a database query and then map the results set to process data so that it can be used to make routing decisions.
- Example 2: A process may map process variables to content metadata, in order to make it persistent in the Documentum repository.
- Example 3: A process may invoke a Web Service. This involves a two-way mapping: first, mapping the process data to input parameters of the Web Service, and second, mapping the output of the Web Service to the process data.

A means for mapping data fields is clearly necessary. Conversely, it is necessary to map process data to a database or other systems. The Process Data Mapper is a graphical tool that simplifies the exchange of process data. Because the mapping is performed graphically, using a simple drag and drop, no coding is required.

In the example in Figure 4, the field “start_date” in the Execution Data object is copied and then mapped to “DateReceived” in Variables.

³ Process data is a general term that encompasses both process variables and content metadata. The latter type is referred to as “packages” in the process.

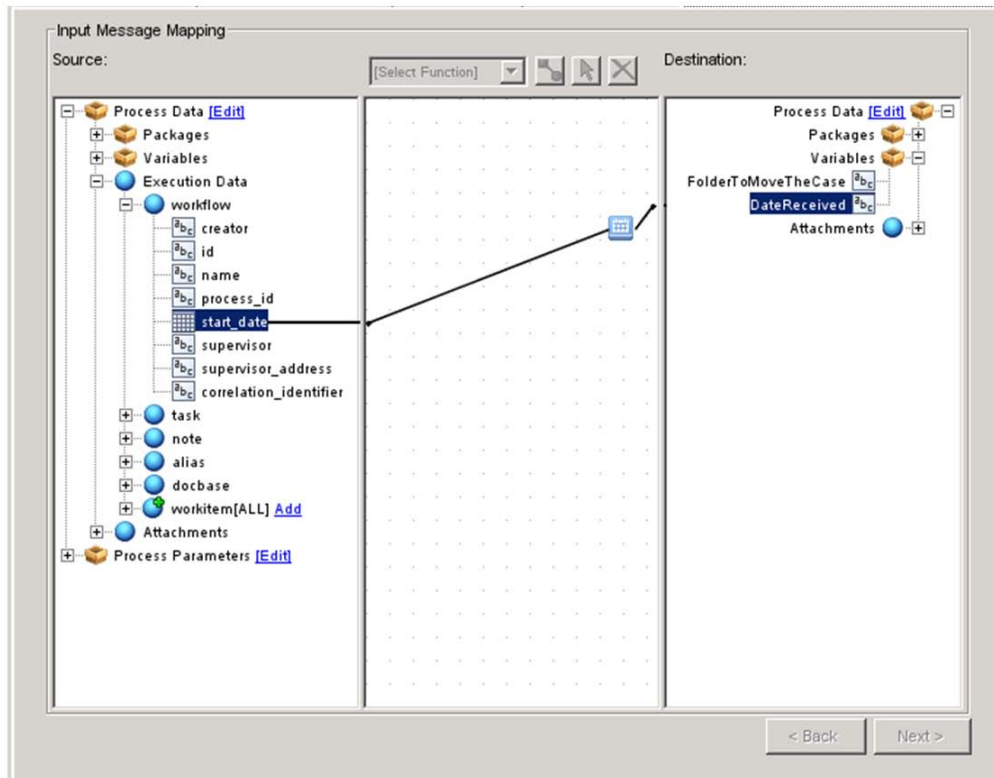


Figure 4. Process Data Mapper

Using the function editor (the drop-down labeled Select Function) the Process Data Mapper can perform complex data transformations as well as simple copying; for example, mapping a date to a string variable (using the date-to-string function). One can also map one variable to multiple variables, as shown in Figure 5.

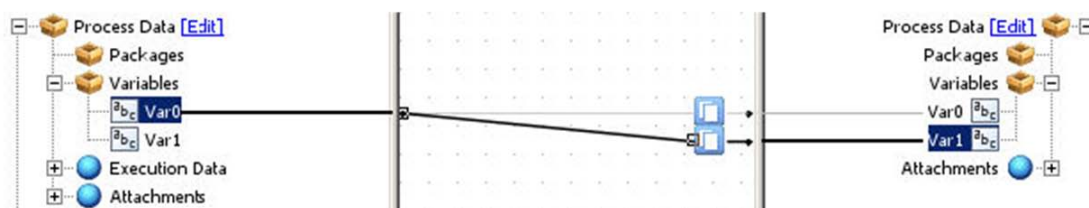


Figure 5. Complex data transformations

Process invocation

A common design pattern is for a process to delegate work. An activity in a process can invoke a service or another process as a composite service. Using an Invoke Process activity template a process invokes a child process. Having invoked the child process, the parent process can either continue its own execution or wait to receive a signal from the child. Using a Post Event to Parent activity template, the child process signals the parent process that it is complete. When the signal is received by the parent, a wait activity will execute and the process continues.

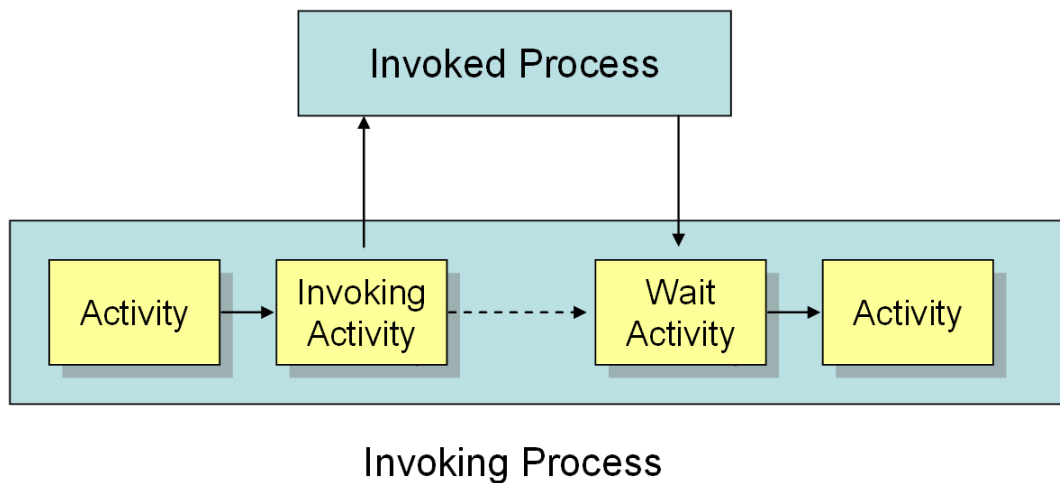


Figure 6. Invoking processes

This begs the question of how the invoked Web Service or process will match its response to the *specific process instance* that invoked it, since there may be multiple process instances in flight at the same time. Correlation solves this problem. To make the match, the request message (from parent to child) will include a correlation identifier that uniquely identifies the parent process instance. The response message (from child to parent) will contain the same correlation identifier. When the Process Integrator⁴ receives the response message, it will use the correlation identifier to route the message to the matching process instance.

User interface patterns

TaskSpace is a highly configurable user interface that unites process, content, and monitoring into a single user experience for transaction-centric business applications. TaskSpace enables you to build and deploy intuitive applications that are tailored to different roles. You can configure applications for task processors who receive and complete work, managers who distribute and track work across their teams, and business owners who need to monitor operational performance in real time. TaskSpace is organized into a series of tabs, in which each tab shows a different view. One of these tabs will display the Task View.

Task View

The Task View presents the actual work to be carried out by the performer and is therefore the most common and important view in TaskSpace. It focuses the user on the information needed to perform the task in the most efficient way possible. In Figure 7 the Task View is shown in the Open Items tab.

⁴ The Process Integrator is the server component responsible for managing integration. It starts processes, manages peer-to-peer and broadcast messaging, invokes services, and handles callbacks.

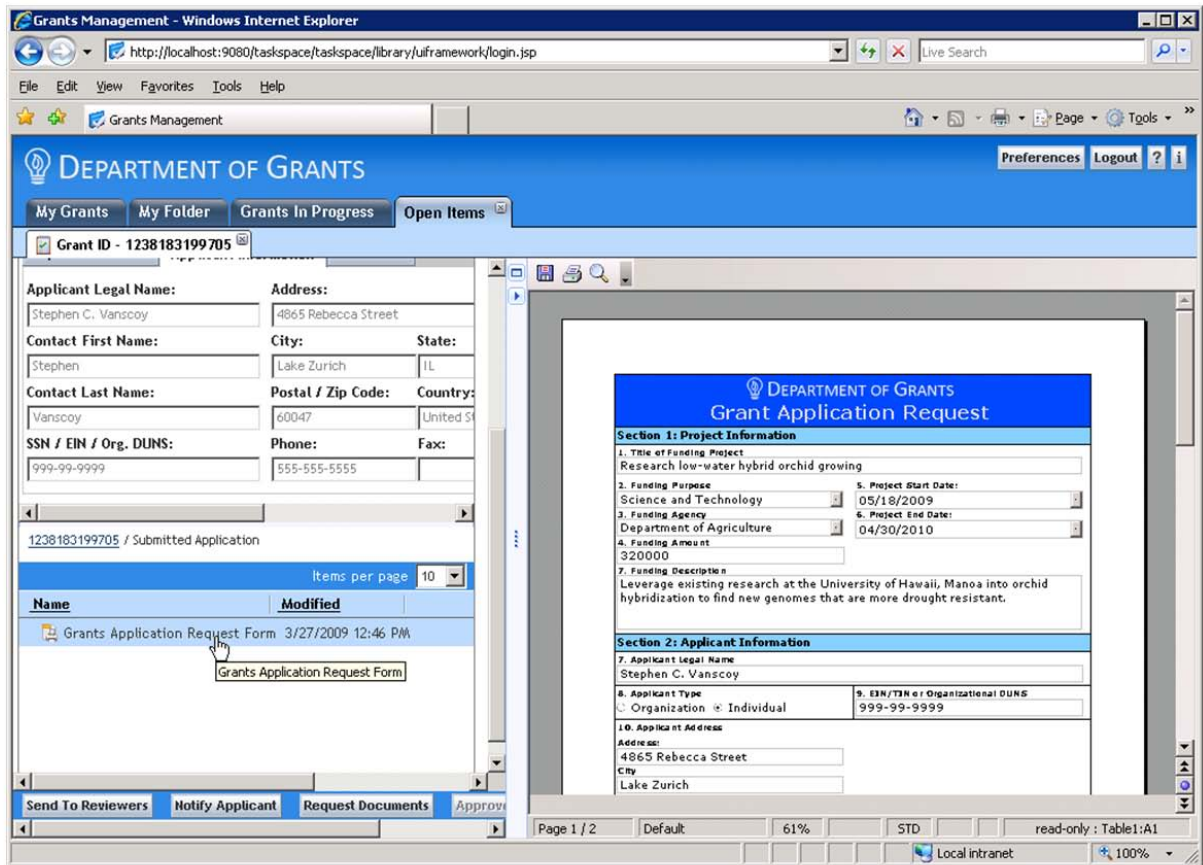


Figure 7. EMC Documentum TaskSpace

This example highlights several important components:

- This shows how a Folder View can be embedded into a Task View
- In the upper left is the metadata, with information about the task
- Under that is a Folder Content View, which lists a set of documents
- To the right is the document preview pane, showing the selected document
- On the very bottom left are several action buttons

The task performer inspects the data, enters any required values, updates data fields, and presses the appropriate action buttons.

Building a Task View is a matter of configuration. The designer selects the process and activity. The designer then drags and drops pre-built UI controls and binds these controls to the process data. The task form in which this is done is shown in Figure 8.

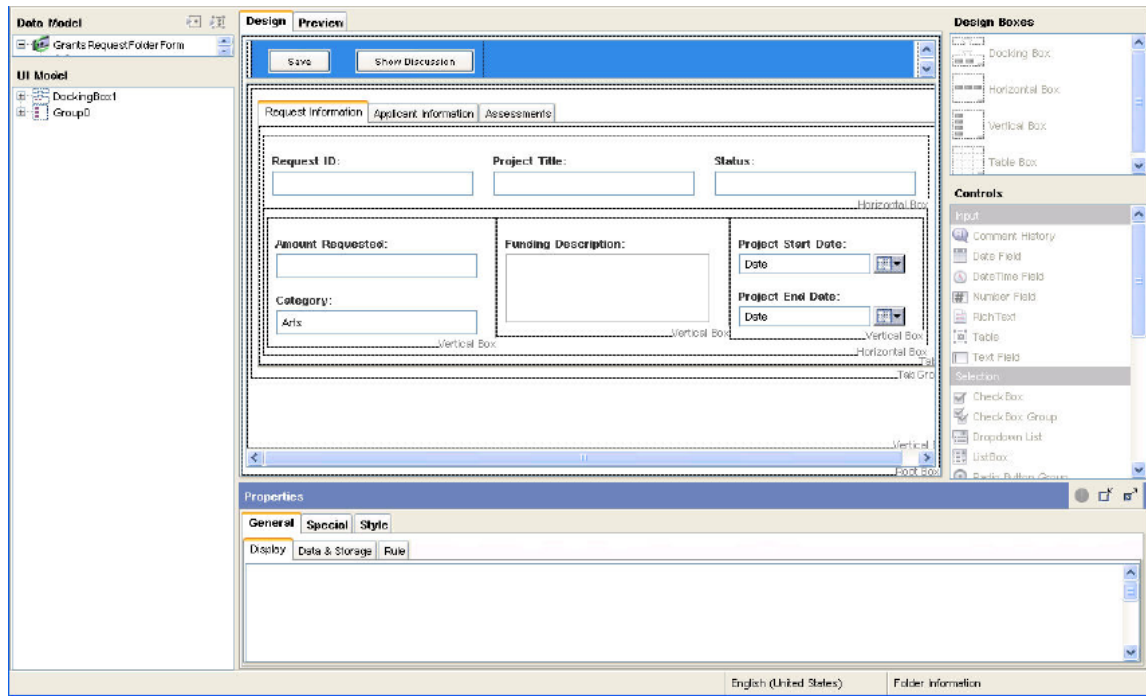


Figure 8. Configuring TaskSpace

Once the form is completed, the designer can set various display parameters in TaskSpace. By configuring pre-built components, a designer can easily create or change the user interface. No coding is needed.

Configurable actions

We have seen several action buttons in TaskSpace. Designers can create action buttons for performing many types of actions; for example, to create documents, forms, or processes. In the example above, the user was presented with the following buttons:



Figure 9. TaskSpace configurable actions

Each of these buttons is used to launch a process. To create an action that will launch a process, the TaskSpace wizard walks the designer through the configuration process. The designer specifies that they want to start a new process, selects the specific process, and finally associates it with a button.

Reporting patterns

In the Documentum Business Activity Monitor (BAM), reports are run against real-time and historical process execution data. These reports are created in a tool called Process Reporting Services (PRS). The report results are displayed in a dashboard, and integrated into a tab in TaskSpace, which provides visual insights into how well the processes are meeting key performance indicators. We look at three important

patterns of reporting: the BAM Control Process, the data source builder, and the dashboard builder.

Unlike other BPM products, xCP's BAM is tightly integrated with process execution, so monitoring processes does not require any complex configuration. Once you execute your process, the BAM data is available and ready for reporting.

BAM Control Process

BAM is designed to monitor the performance of individual processes, using the information it extracts from the Process Engine. BAM can focus on individual process instances, to show exactly how the process has progressed and which activity is executing at the present moment. However, in complex business applications in which many business processes may be executing in parallel, it is valuable to provide a central point of visibility and control. This goal is achieved by the BAM Control Process, a single process that collects information from the other processes and provides a consolidated view. Managers and other users can then see key milestones and understand the status of the overall application. Figure 10 is a typical example of a BAM Control Process used in a Grants Management application:

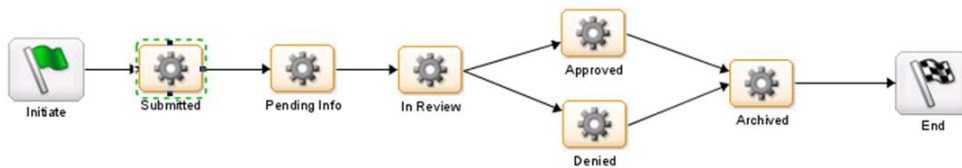


Figure 10. BAM Control Process

Each activity in the BAM Control Process is an automated activity. Activities in the individual processes signal these activities to indicate that they are in progress.

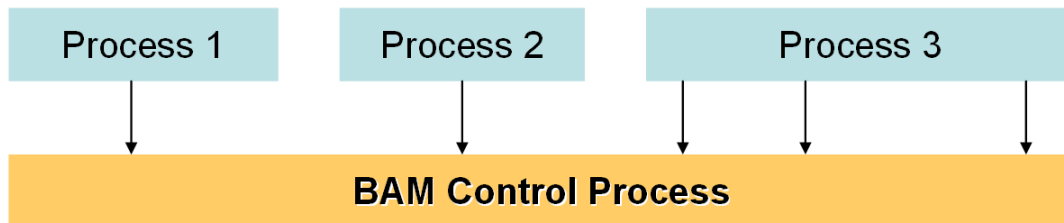


Figure 11. Individual processes signal a BAM Control Process

Naturally, the inter-process communication needs to convey the case identifier so that the individual process instances can be correlated into a single BAM Control Process instance.

With this information, the BAM system can run various reports and populate a dashboard showing the state of the application.

Data source builder

Reports are logically divided into two parts: the data source definition and the specification of the report format. The data source defines the logical structure of the

data. PRS enables the report designer to define data sources without programming or knowledge of SQL. Instead, the user simply drags and drops report entities.

Figure 12 is an example of a data source with a parent entity and a child entity.

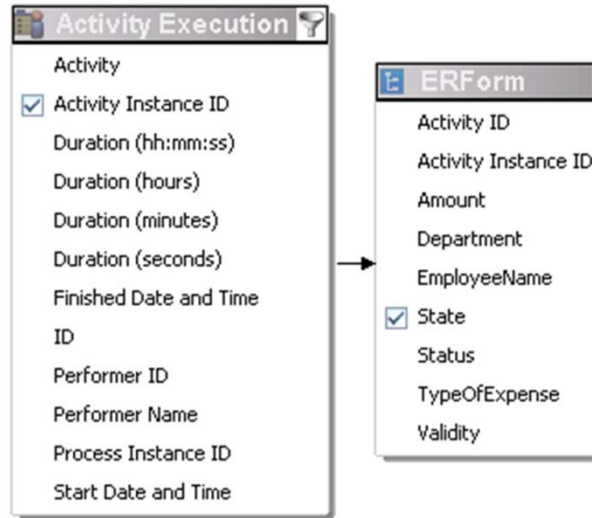


Figure 12. Data source with a parent entity and child entity

The parent entity, Activity Execution, corresponds to activity instances. The child entity, ERForm, refers to process data. In this example it is an Expense Report that the user fills out. The designer builds the data source by dragging and dropping entities from a context-sensitive palette. Each element of this report will be an ordered pair (Activity Instance ID, State), as shown in Figure 13.

The screenshot shows a 'Data Source Preview' window with a table containing the following data:

Activity Instance ID	State
4a00000c80011d1c	NY
4a00000c80011d1d	MA
4a00000c80011d1e	CA
4a00000c80011d1f	MA
4a00000c80011d20	MA
4a00000c80011d30	NY
4a00000c80011d33	NJ

At the bottom of the window, there is a 'Limit Rows' checkbox checked and a value of 100.

Figure 13. Data source items leveraged in reports

Case Management example

Having described the major technology aspects of xCP, we now turn to a class of business problems for which it is well suited. Case Management is an increasingly important class of applications, with relevance to many industries, including public sector, healthcare, insurance, financial services, and law. The case is the central point of control that contains documents, tasks, data, and discussions. The case can be accessed and updated by a set of business processes. In the course of these processes, caseworkers update documents, add documents, and enter information. This information enables case workers and case managers to make decisions about its disposition. The case persists even after the processes have run their course. This ensures that cases can always be audited and that future cases can be linked to prior cases.

Personas

As a special case of Case Management we will look at Grants Management. The approach and technical features of this special case apply generally to Case Management applications. In our example there are three protagonists:

- Applicant, an investigator seeking funding from a government agency
- Grants Manager, an employee of the government agency who handles the application (that is, the “case”) from start to finish
- Reviewer, an expert who evaluates the grants proposal in terms of importance, ability to execute, appropriateness of the funding, and so forth

Application narrative

The personas interact in the following way:

1. An Applicant submits a grant application form.
2. The Grants Manager receives the application and checks it for validity.
3. The Grants Manager may request the Applicant to forward additional information.
4. The Grants Manager sends the Grant Request to Reviewers, who make comments and submit their formal votes.
5. Based on their votes, the Grants Manager approves (or denies) the application and then notifies the Applicant.

Reusable components

The underlying case object is the Grant Request folder, which contains two sub-folders:

- Grant Submitted Application, which holds the form received from the Applicant
- Grant Review History, which holds information on the reviews

Creating a new case is a simple matter, since the Process Builder⁵ comes with a pre-built activity template that creates new cases automatically based on the form template.

There are two major persistent Documentum objects created in the application:

- Grants Application Package, essentially holding the form filled out by the applicant
- Grants Request Package, which contains all the information created in the lifecycle of the grant

Processes

The Grants Management application makes use of many processes to accomplish its work. Two of these involve users, and the rest are “utility” processes. The logical relation between the processes is shown in Figure 16.

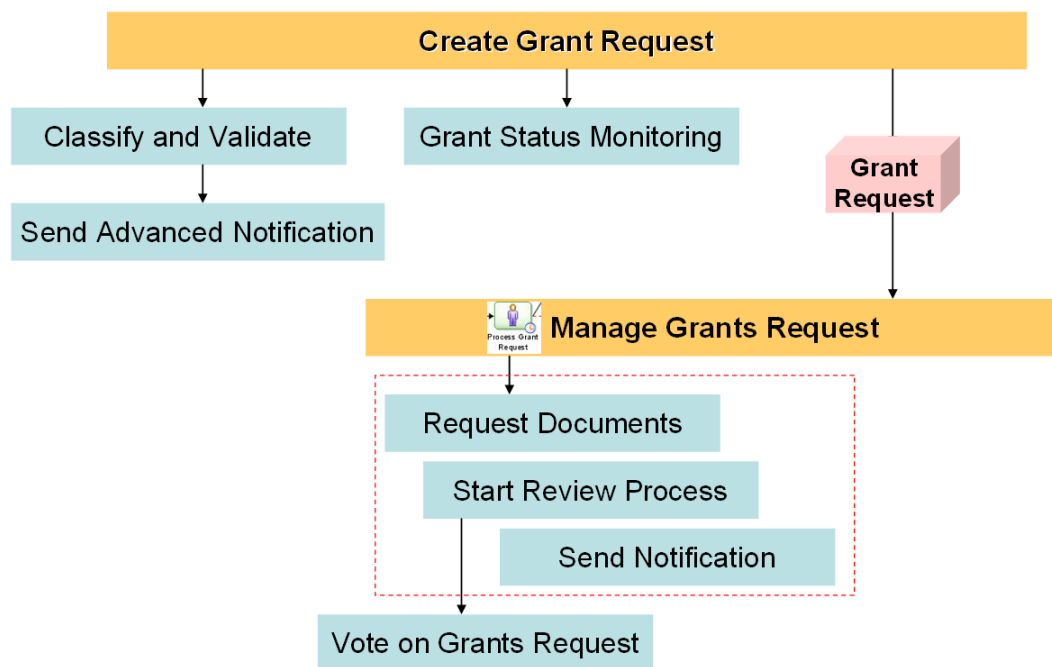


Figure 16. High-level overview of the Grants Management process

The outcome of the Create Grant Request process is to create a new Grant Request object. Once this object has been created, a Manage Grants Request process is automatically initiated. The main control process is the one shown in the center, “Manage Grants Request.” It includes a single manual activity “Process Grant Request,” in which the Grants Manager is given a choice of actions, each of which triggers another process: Request Documents, Start Review Process, and Send Notification (inside the red box). All other processes in the diagram are invoked from

⁵ The basic design tool for processing services and handling callbacks.

their parent processes; for example, Classify and Validate is invoked by Create Grant Request.

Anatomy of a process

We will take a closer look at one of the processes in the Grants Management application, namely Manage Grants Request.

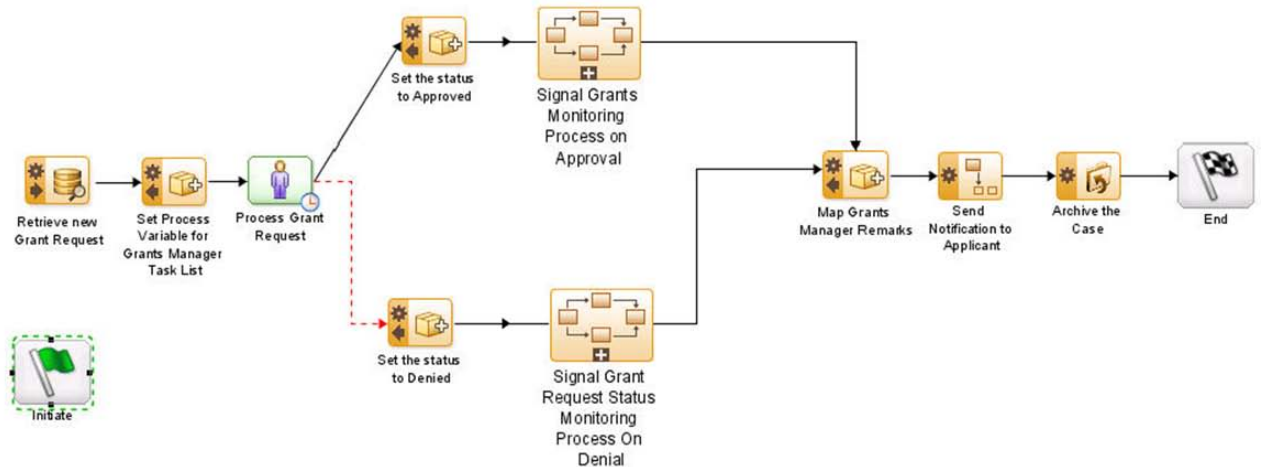


Figure 17. Manage Grants Request process

The process begins with an automatic initiate activity that retrieves the next grant request from the Documentum Repository. The second activity is process data mapping, which maps attributes of the Grant Request object to process data. The third activity is a manual one – a human task, in which the Grants Manager takes action on this request. We have already seen the Task View, as shown in Figure 18.

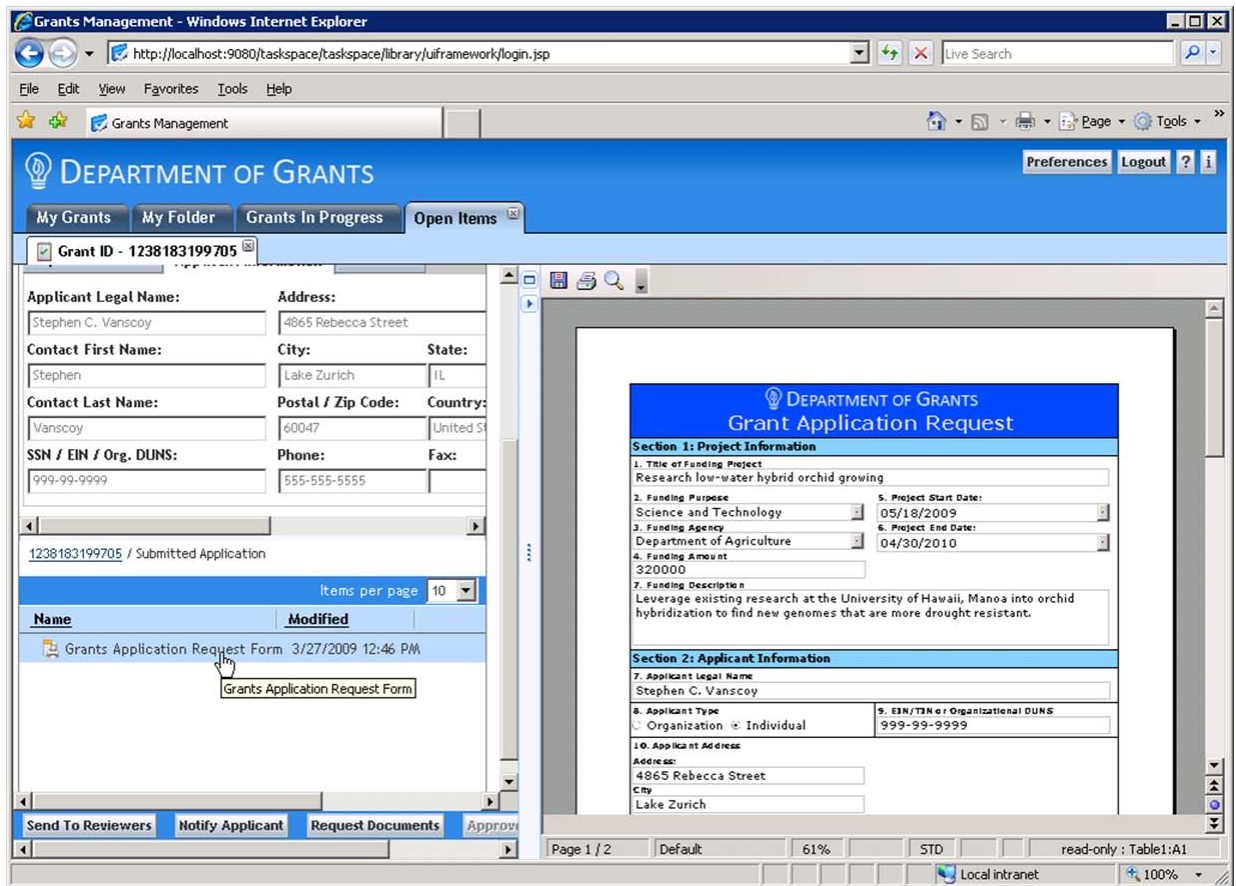


Figure 18. Grants Managers user interface

The Grants Manager can choose to click any of these action buttons. When the Grants Manager clicks the Send To Reviewers button, several things will happen. First, it will trigger a forms adapter, which retrieves the list of potential reviewers from the Documentum Repository. This is used to populate the form shown in Figure 19, in which the Grants Manager selects the actual reviewers.

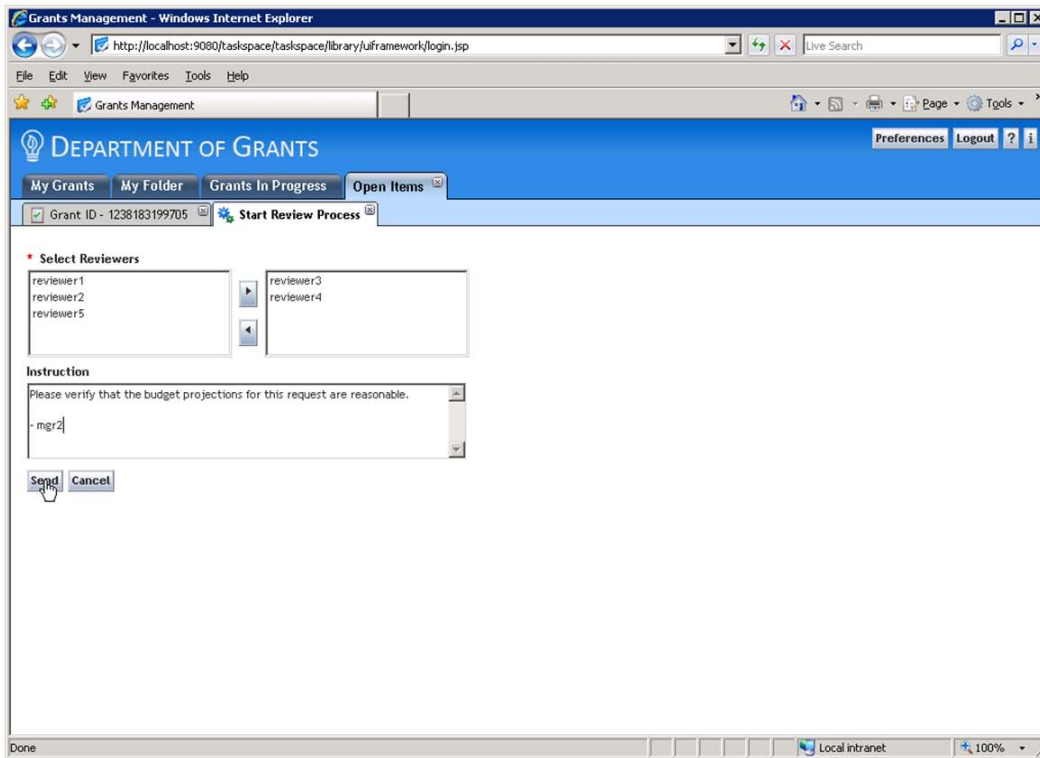


Figure 19. Assign Reviewers interface

Once the user clicks the Send button, the Start Review Process will be launched and the information on this form (the list of reviewers) will be passed to the process.

As shown in Figure 20, there are three utility processes available to the Grants Manager. Each one is triggered by a corresponding action button.

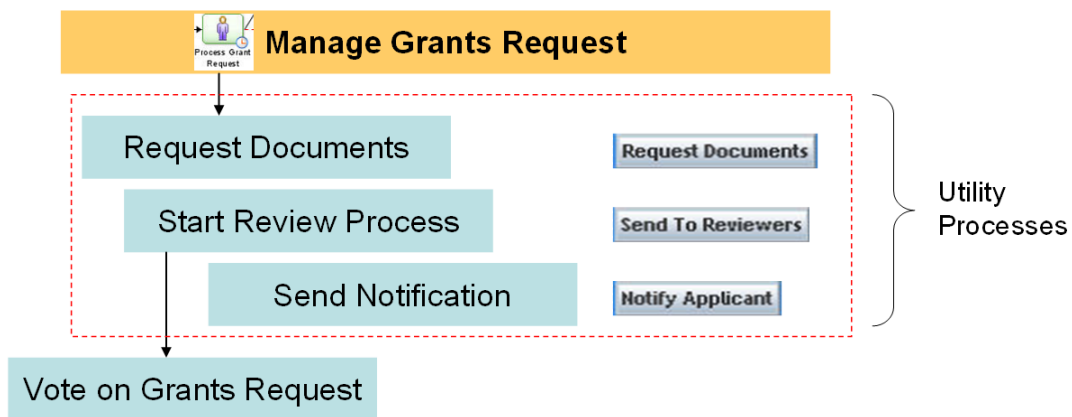


Figure 20. Triggers within the Grants Request process

We also see another process invocation. An activity in the Start Review Process will invoke the Vote on Grants Request process in which the reviewer fills out a review form and recommends approving or denying the grant request.

The remaining activities in Manage Grants Request send data to the monitoring process (needed for the BAM dashboard), map the reviewers' comments to the Grant Request object, send e-mail to the applicant informing them that their grant was approved, and finally archive the grant.

The “four stroke engine”

The Grants Management application includes a solution pattern that reoccurs over and over.

1. The task performer decides to take an action and clicks an action button.
2. Clicking the button launches a form, allowing the user to enter data.
3. A form adapter may also be called (as in our previous example, where it was needed to populate the list of reviewers).
4. The action button then triggers a business process, which consumes the process data from the form.

This “four stroke engine” pattern can be called recursively, since the new process can present the user with another task with action buttons, thereby invoking another process. In general, the xCP architecture is based on a fractal approach in which structures “can be split into parts, each of which is [at least approximately] a reduced-size copy of the whole.”

This pattern gives the Grants Manager the maximum amount of freedom to apply expertise and make decisions in an ad hoc way. At the same time it allows utility processes to be invoked, which carry out many steps in an automated, structured, repeatable way.

Conclusion

In this paper, we have seen that Documentum xCP applications are created by configuring pre-built components and assembling them according to documented design patterns. We looked at several of the components and design patterns in the context of a case-based application. We now consider the benefits that result from such an approach.

Cost reduction

The cost of development is of acute importance to both user organizations and systems integrators. With the xCP approach, based on pre-built components and design patterns, a development project will require fewer expensive resources, such as expert Java programmers. The effect is to reduce development costs considerably, which will translate into greater ROI for the user organization, and higher margins or a better win ratio for the systems integrator. If we consider the entire lifecycle of the application, including modifications, maintenance, and upgrades, it will reduce the total cost of ownership (TCO), which tends to be much greater than the initial development costs.

Risk avoidance

Custom development is inherently risky. Interfaces can be complex and changes made to one area of the code can have unpredictable effects on other areas. This means frequent design reviews and many levels of software testing. An advantage of using proven, tested design patterns is to minimize technical risk. By following best practices, potential problems with performance, usability, and agility can be averted, thereby eliminating scrap and rework.

Faster time to market

In many IT projects, the most critical goal is speed to business results; delivering a solution quickly provides a competitive advantage. By assembling solutions from components, development can be accelerated substantially, as much as twice as fast compared to traditional hand-crafted code. In some cases the acceleration factor will be even greater.

Ease of extensibility

As experience is gained in the use of the xCP components, your application builders will learn how to solve a wide class of problems. New solution patterns will emerge and be compiled into libraries of best practices. All this will lead to greater efficiency and effectiveness. It will enable the framework to be extended into a variety of case-based and process-centric solutions, across many application areas, such as financial systems, account management, policy governance, R&D, service requests, employee grievances, and contracts.

About EMC

EMC Corporation (NYSE: EMC) is the world's leading developer and provider of information infrastructure technology and solutions that enable organizations of all sizes to transform the way they compete and create value from their information. Information about EMC's products and services can be found at www.EMC.com.

Take the next step

To learn more about how your organization can benefit from an application built on EMC Documentum xCP, visit <http://www.EMC.com/xcp> or call 800.607.9546 (outside the U.S.: +1.925.600.5802).

Visit the Documentum Developer Community at <http://developer.emc.com/documentum> for technical product information, support, tutorials, sample code, documentation and best practices.