

tutorial

Describing and Using Patterns for UI Design

Daniel Lafrenière, TELUS
Åsa Hedenskog, Ericsson

© 2001 by Daniel Lafrenière and Åsa Hedenskog
All rights reserved



Abstract

In this tutorial, participants will learn the background of patterns, and experiment with the discovery, description and application of patterns through hands-on experience. The exercises will be based on our *Pattern Supported Approach to the User Interface Design Process (PSA)*, since it spans many aspects of pattern usage and, most importantly, will provide a common ground for tutorial participants.

Since patterns are best understood by seeing and using, the emphasis will be on the exercises. Our goal is to give the participants a good feeling for what patterns are about, and how they can be used for strengthening the design process.

This tutorial has been taught during UPA 2000, TorCHI 2000, to a group of usability consultants in 1999 (in Sweden), to a group of researchers and practitioners at the Ericsson Conference of Usability Engineering 1999, and at Laval University in Québec, QC at the graduate level in multimedia in 1999 and 2000. A workshop was also organized during UPA 1999 in Scottsdale, AZ with great success.

This latest version is especially created for HCI practitioners, by emphasizing the practical aspects of patterns and how to integrate them into participants' own work practice.

Attendees will gain an understanding of issues in user interface patterns:

- Understanding the basics of patterns in the UI design process
- How to discover, describe and use patterns
- How to integrate patterns in your own practice



Instructor Biographies (2005 update)

Daniel Lafrenière (daniel.lafreniere@gmail.com) is a consultant in UI methodologies, architecture and design at TELUS. His contracts include task analysis, writing guidelines, UI architecture and design, evaluation, and integrating user-centered techniques within the software development lifecycle. He published the book *Créez des interfaces gagnantes (Create Winning User Interfaces)*, which is a synthesis of techniques and guidelines on user interface design. In 1993 he co-authored a course on HCI for the Computer Science bachelor program at Université Laval. He has been teaching HCI there since, in the CS Department and more recently in the Visual Arts School (masters in multimedia). Daniel is a member of ACM SIGCHI and UPA.

Åsa Hedenskog (asagr@ida.liu.se) is a usability engineer at Ericsson Radio Systems AB. She works with support for Ericsson's method for user centered design - DELTA. Her work at Ericsson also includes covering new tools and techniques for usability engineering, participating in applied research projects aimed at identifying and evaluating new tools and techniques, method development (DELTA) and teaching. Previous to joining Ericsson, Åsa worked as a usability consultant doing task analysis, user profiling, prototyping, testing, evaluation and process development. The work also included teaching, introducing usability in client organizations and giving method support. Åsa is a member of ACM SIGCHI, UPA and has been president of the Linköping Local Chapter of UPA (Sweden).



Agenda

| | |
|-------------|--|
| 08:30-10:00 | Introduction and Background of Patterns |
| 10:00-10:30 | Break |
| 10:30-11:15 | Hands-on Session # 1 : Using Design Patterns |
| 11:15-12:00 | The Pattern-Supported Approach to the UI Design Process |
| 12:00-13:00 | Lunch |
| 13:00-13:30 | The Pattern-Supported Approach to the UI Design Process (cont'd) |
| 13:30-14:15 | Hands-on Session # 2 : Preparing for User & Task Analysis |
| 14:15-14:45 | Capturing Patterns |
| 14:45-15:00 | Hands-on Session # 3 : Describing a Task Pattern (first half) |
| 15:00-15:30 | Break |
| 15:30-17:00 | Hands-on Session # 3 : Describing a Task Pattern (second half) |
| 17:00-17:30 | Plenary Discussion: Participants' Experiences and Conclusion |



Objectives of the Tutorial

- at the end of the tutorial, you will:
 - understand the basics of patterns in the UI design process
 - understand about the discovery, description and usage of patterns
 - get started on thinking of how to integrate patterns into your own practice
- the focus of this tutorial will be on task analysis and design



Part 1 Introduction



Introduction

- there is a need to **document experience**
- there is a need to communicate between people that are involved in the UI design process (usability engineers, designers **and** users)
- we think that patterns offer support both for documenting design knowledge and for facilitating communication



What is a Pattern ?

A pattern is a **formalized** description of a **proven** concept that expresses non trivial **solutions** to a UI design **problem**. The primary goal of patterns in general is to create an inventory of solutions to help UI designers resolve UI development problems that are common, difficult and frequently encountered.

— adapted from
[Loureiro & Plummer, 1999]

[Patterns provide] ...possible good **solutions** to a common design **problem** within a certain **context**, by describing the invariant qualities of all those solutions.

— [Tidwell, 1999]

A pattern is a named nugget of insight that conveys the essence of a **proven solution** to a **recurring problem** within a certain **context** amidst **competing concerns**.

— [Appleton]



What do these Definitions Have in Common?

- a **proven**, **generic** solution to a **recurring** problem
- the problem occurs within a certain **context**
- there are numerous **competing concerns** present
- we also want a **formalized** description

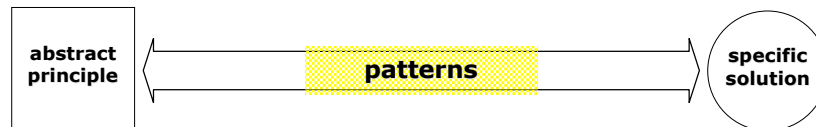


What is Proven?

- a **real** solution, **not** just a strategy or an abstract principle



Why Generic?



Why Recurring?

- we have to make sure that we're **not** producing theories or speculations - *Rule of Three*



Why Context?

- we need to know when the solution is applicable



Why Competing Concerns (Forces)?

- why isn't the solution trivial?
- the proposed solution balances the concerns in the best manner for the given context



Simply Put ...

- patterns can be said to provide powerful and generic design **guidance** at a format that is consistent and easy to read and understand – they convey knowledge about good design



Format

- name
- problem
- context
- forces
- solution
- examples
- resulting context
- rationale, related patterns and known uses



Name

- meaningful
- refers to the knowledge it describes (sometimes AKAs)

example:

- Progress indicator
(adapted from [Tidwell, 1999])



Context

- describes the **user's** goal

example:

- A time-consuming process is going on, the results of which is of interest to the user.



Problem

- describes the **design** goal
- what is the intent of solving the problem?
- what are the objectives?
- what are we trying to accomplish?

example:

- How can the artifact show its current state to the user, so that the user can best understand what is going on and act on that knowledge?



Forces

- what are the relevant concerns when solving a problem?
- what forces and constraints apply, and how do they interact/conflict (with one another or with the goal we wish to reach)?
- the solution will have to accommodate trade-offs between the forces

example:

- The user wants to know how long they have to wait for the process to end.
- The user wants to know that progress is actually being made, and that the process hasn't just *hung*.
- The user wants to know how fast the progress is being made, especially if the speed varies.
- Sometimes it's impossible for the artifact to tell how long the process is going to take.



Solution

- a description of how to realize the desired outcome

example:

- Show the user a status display of some kind, indicating how far along the process is in real time. If the expected end time is known, or some other relevant quantity (such as the size of a file being downloaded), then always show what proportion of the process has been finished so far, so the user can estimate how much time is left. If no quantities are known - just that the process may take a while - then simply show some indicator that it's still going on.



Solution (cont'd)

- a description of how to realize the desired outcome

example (cont'd):

- Animation is often used to good effect in this pattern; motion draws the user's attention, and its cessation implies a new relaxed, stable state of being (*the process is done, so you can relax now*).
- Sound can also be used this way, for the same reason. Be subtle, though, and be aware of the importance of the process relative to the other things demanding the user's attention.



Examples

- help a reader understand if a pattern is applicable, and how to use it
- describe an initial context, how the pattern is applied to a specific problem, and what the resulting context is

example:

- Countdown timer on a microwave oven.
- The percent-complete message during a file download.



Part 2

Background



A Bit of History

- architect Christopher Alexander wanted to create structures that improve people's comfort and quality of life
- he promoted *timeless design* ideas to make this happen
- he proposed a paradigm for architecture based on three concepts:
 - the **Quality** (*the Quality Without a Name*)
 - the **Gate**
 - the **Way** (*the Timeless Way*)



The Quality

- this is the **essence** of all things living and useful that imparts unto them qualities such as: freedom, wholeness, completeness, comfort, harmony, habitability, durability, openness, resilience, variability, and adaptability
- it is what makes us feel alive and sated, gives us satisfaction, and ultimately improves the human condition



The Gate

- this is the mechanism that allows us to reach the quality
- it is manifested as a living common pattern language that permits us to create multiform designs which fulfill multifaceted needs
- it is the universal ether of patterns and their relationships that permeate a given domain
- the gate is the conduit to the quality



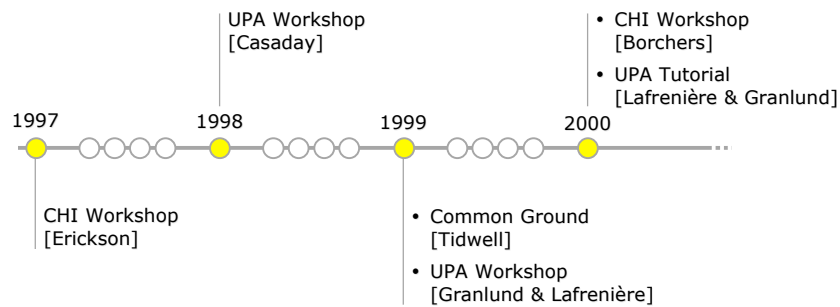
The Way

- using the way, patterns from the gate are applied using a technique of differentiating space in an ordered sequence of piecemeal growth: progressively evolving an initial architecture, which then flourishes into a live design possessing the quality
- by following the way, one may pass through the gate to reach the quality



HCI Patterns

- are being used implicitly by skilled UI designers ... however, these designers usually keep little in the way of formal (or documented) descriptions of these solutions
- are quite recent:



HCI Patterns vs Style Guides

patterns

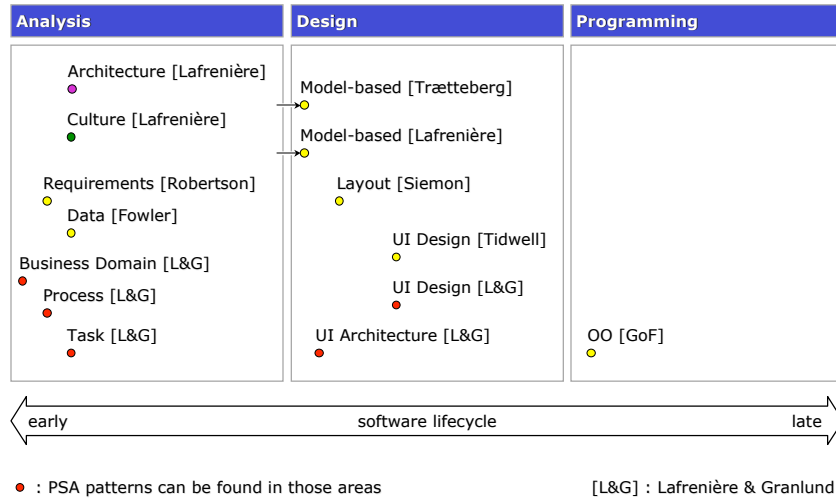
- provide a solution to a re-occurring problem in a specific context
- carry usability information about the user, the context and the task
- usually point to other patterns in a network way, forming a family of inter-related patterns

style guides

- specify graphical appearance and behavior in accordance with a manufacturer's look & feel
- don't provide contextual information ... more preoccupied by the microscopic aspect of design



Patterns in the Software Lifecycle



Architecture Patterns [Alexander & al., 1977]

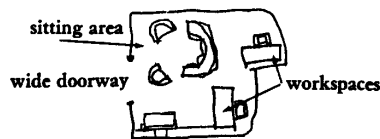
Half-Private Office (Pattern # 152)



What is the right balance between privacy and connection in office work ?

The totally private office has a devastating effect on the flow of human relationship within a work group, and entrenches the ugly quality of office hierarchies. At the same time, there are moments when privacy is essential; and to some extent nearly every job of work needs to be free from random interruption ...

Avoid closed off, separate, or private offices. Make every workroom, whether it is for a group of two or three people or for one person, half-open to the other workgroups and the world immediately beyond it. At the front, just inside the door, make comfortable sitting space, with the actual workspace(s) away from the door and further back.



Musical Design Patterns [Borchers]

- **Name:** Triplet Groove
- **Context:** Playing music in the *Jazz Style*.



- **Forces:** Players need to create a swinging feeling that the straight rhythm from other musical styles does not convey. But: Sheet music cannot include all rhythmic variances; it would become unreadable.
- **Solution:** Where the score contains an evenly spaced pattern of eighth notes, shift every second eighth note backwards in time by about one third of its length, shorten it accordingly, and make the preceding eighth note one third longer. The length ratio changes from 1:1 to 2/3:1/3. Two straight eighth notes become a triplet quarter and eighth note. The result is a *laid-back groove*.
- **Examples:** Any recorded Jazz piece features this rhythmic shift. The actual shift percentage varies widely: usually, the faster a piece, the less shifting takes place.
- **Consequences:** This pattern uses an underlying straight beat like *4/4 Rhythm*.



Why Patterns?

- consistent, easy to read, provide *background reasoning*
- accessible (**easy** to read and evaluate)
- *lingua franca* (regardless of background and experience) for usability engineers, designers **and** users
- allows *cross-fertilizing*
- promotes **reuse**
- using patterns should be like asking your experienced colleague in the next room
- we could have used something else but we think patterns represent a very good way of passing on knowledge



[Borchers, 2000] Claims

- HCI patterns must be readable and understandable by professionals and non-professionals alike (which is not the case in Requirements and OO Patterns)
- HCI patterns will take power from HCI designers and put it into the hands of users
- the use of patterns in the various domains can be mapped to most phases of the usability engineering lifecycle



Hands-on 1

Using Design Patterns

30 min.



Part 3

The PSA Approach



The PSA Approach

- patterns describe generic solutions to common problems in context
- up to this point, most of the work on patterns for UI has focused on screen design issues
- PSA suggests a **wider** scope for the use of patterns by looking at the overall user oriented UI design process

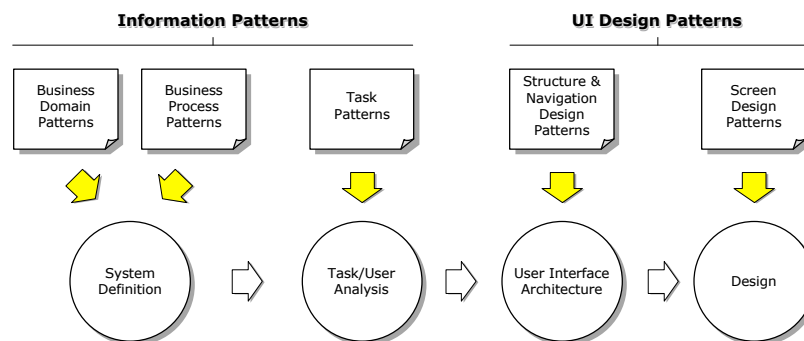


Our View on UI Patterns

- we think patterns will be a valuable source of information, supporting both the analysis of the current situation and the design of the new system
- we do **not** think of patterns as the sole source of information for documenting and passing on design knowledge — it is an **additional** source with certain valuable qualities

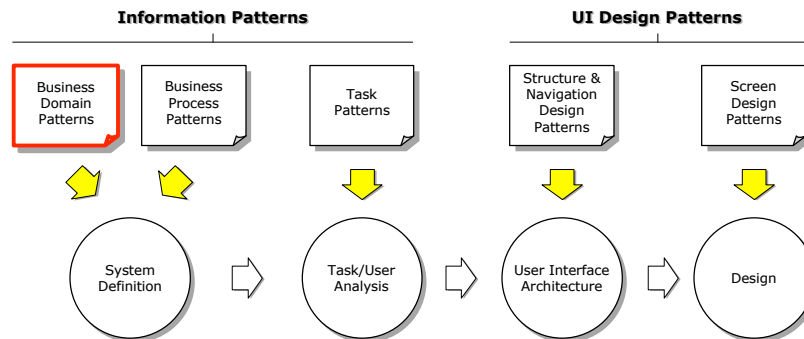


PSA Overview



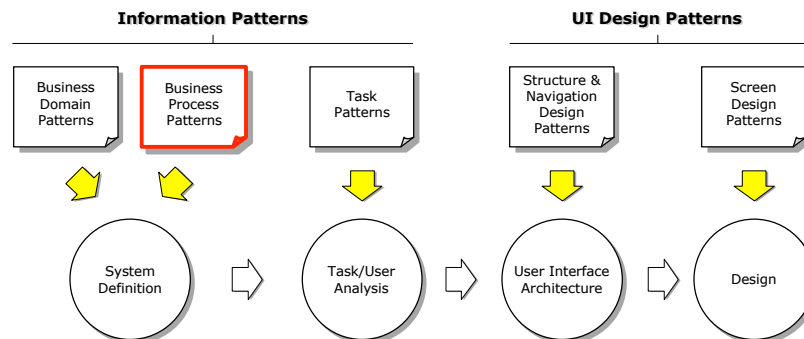
Business Domain Patterns

- describe the type of business, its goals, and the actors and business processes typically involved



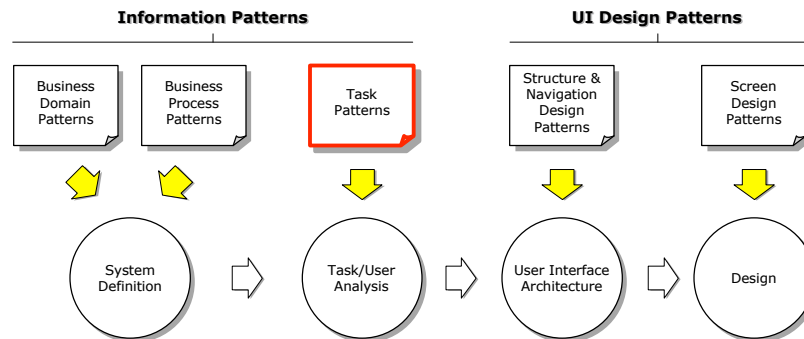
Business Process Patterns

- describe typical processes and actors involved in the delivery of services/goods in compliance with business goals



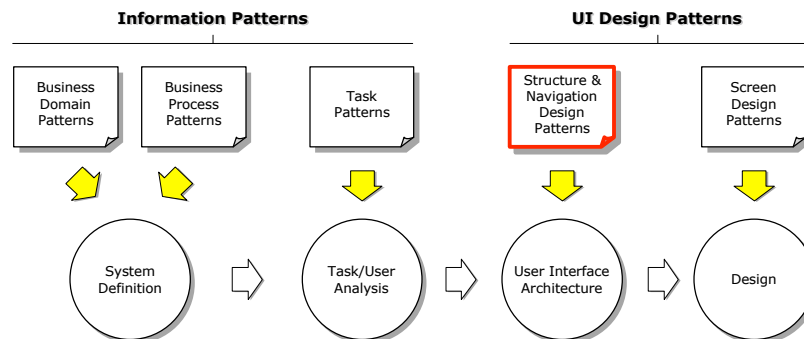
Task Patterns

- capture and pass on knowledge about typical users, the work context and the task



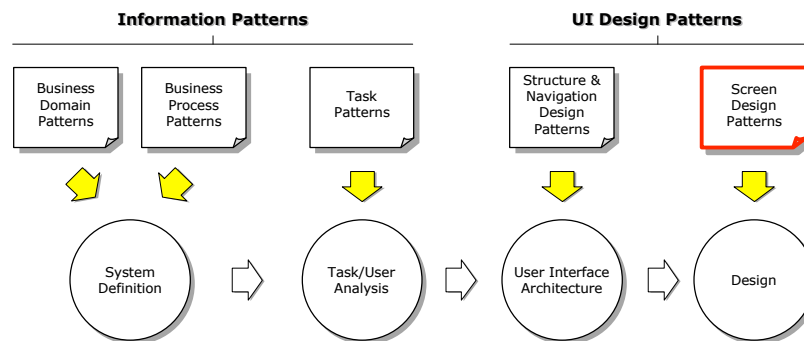
Structure & Navigation Design Patterns

- describe ways to structure information and/or functions based on navigational styles, in order to support the user's task



Screen Design Patterns

- document GUI design issues



PSA Patterns

- information patterns (**domain specific - non generative**):
 - business domain
 - business process
 - task
- design patterns (**generic patterns - generative**):
 - structure & navigation design
 - screen design



Why Information Patterns?

- we do reuse information
- you don't have to completely re-analyze a task that has already been analyzed many times

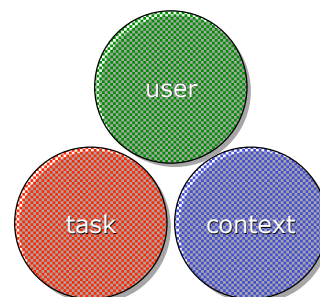


- compiled and accessible information
- provides earlier and more relevant feedback from users ... those information patterns can be used as the starting point for requirement analysis
- ref. to [Robertson, <http://>] requirements patterns



PSA Patterns

- the usability of a system emerges as the product of the following:
 - user
 - task
 - context of use
- learning about these three factors, and designing according to this knowledge, is the key to successful system design from a usability point of view

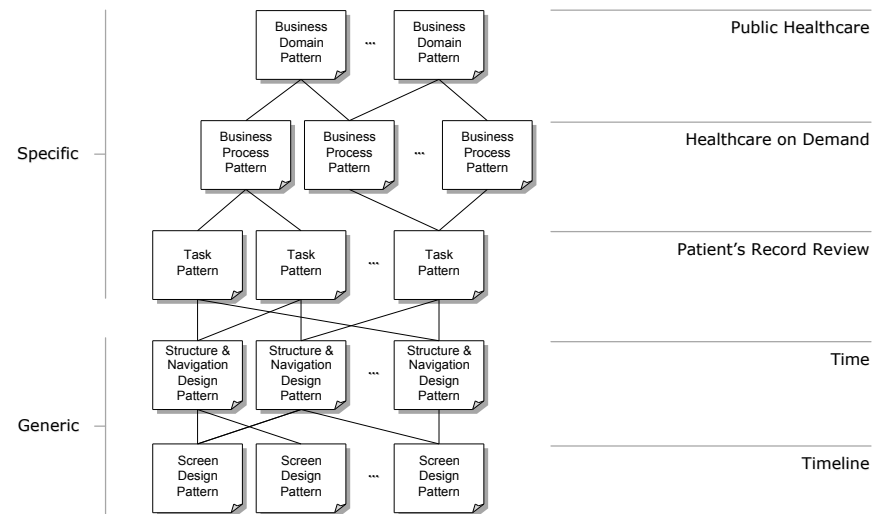


What is PSA for?

- to strengthen the analysis of the user, the context and the task by drawing upon and communicating knowledge from previous design projects thus avoiding re-analyzing problems and re-compiling information
- to guide and support the usability team throughout the design process



PSA Patterns



Description Format

- strongly based on the **Alexandrian** form:
 - name
 - context
 - problem
 - forces
 - solution
 - example
- *all* elements are **not** used for *all* pattern descriptions *all* the time



Business Domain Pattern

- describes the type of business, its goals, the actors and business processes typically involved
- acts as a starting point
- helps communicate the system *vision*



Business Domain Pattern Example

- **Name:** Public Healthcare
- **Context:** The business is to provide good healthcare to the public audience, with governmental funding.
- **Forces:**
 - The audience is very large
 - Monetary funding is limited
 - There is personnel shortage (doctors, nurses and administrators)
 - Service delivery time is always crucial
- **Resulting Business Process patterns:**
 - Preventive healthcare
 - Emergency healthcare
 - Healthcare on demand (patient looks up the service due to some perceived health problem)



Business Process Pattern

- describes typical processes and actors involved in the delivery of services/goods in compliance with the business goals
- narrows down the system definition and points to specific *task* patterns



Business Process Pattern Example

- **Name:** Healthcare on Demand
- **Context:** The business provides care to patients who seek help because they are experiencing a health problem (that is not of an emergency type).
- **Forces:**
 - A great number of people seek help (want doctor's appointments)
 - Monetary funding is limited
 - There is a personnel shortage (doctors, nurses and administrators)
 - Service delivery time is always crucial
 - Doctors are not the same from one time to another
 - Patient's record must be available to all doctors
 - Some problems that patients seek help for turn out to be emergencies
 - Some patients seek help without really needing health care
- **Resulting Task patterns:**
 - Appointment Taking
 - Patient Registration
 - Patient Examination
 - Patient's Record Review
 - Patient's Record Update
 - Medication Prescription



Task Pattern

- describes the goal of the task, typical users, work context and task
- points to *structure & navigation design* patterns
- may contain sub-task patterns



Task Pattern Example

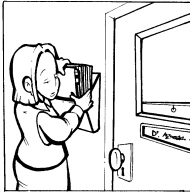
the pattern has a name, describing the task

this pattern is a specific instantiation of the generic pattern "Case Folder Analysis" which can apply to many different domains

- **Name:** Patient's Record Review
- **Context:** A doctor wants to review a patient's record before talking with him/her in her office (she has few minutes to do this). The record will typically include time related information about illnesses, exams and medications.
- **Problem:** How can we design the interface to provide an overview and at the same time enable comparison and correlation of data, and pattern detection.
- **Example:**



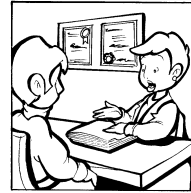
The medical secretary locates the medical record.



She leaves the folder for the doctor to pick up.



The doctor picks up the medical record and reads it rapidly to get an overview ... she usually has less than one minute.



When the patient has arrived, the doctor asks what the problem is.



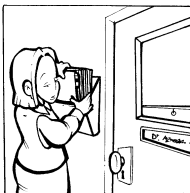
Task Pattern Example (cont'd)

describes the user's goal

- **Name:** Patient's Record Review
- **Context:** A doctor wants to review a patient's record before talking with him/her in her office (she has few minutes to do this). The record will typically include time related information about illnesses, exams and medications.
- **Problem:** How can we design the interface to provide an overview and at the same time enable comparison and correlation of data, and pattern detection.
- **Example:**



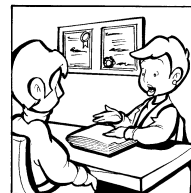
The medical secretary locates the medical record.



She leaves the folder for the doctor to pick up.



The doctor picks up the medical record and reads it rapidly to get an overview ... she usually has less than one minute.

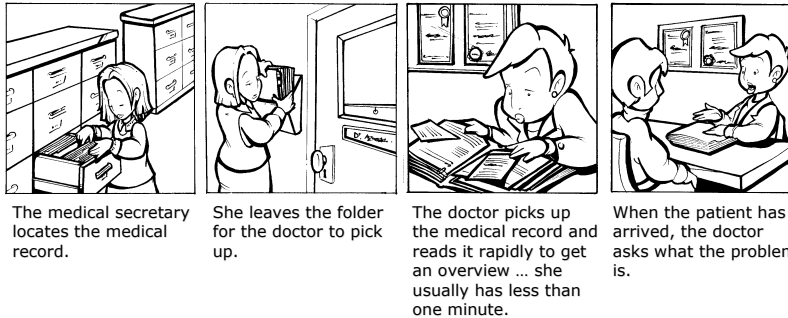


When the patient has arrived, the doctor asks what the problem is.



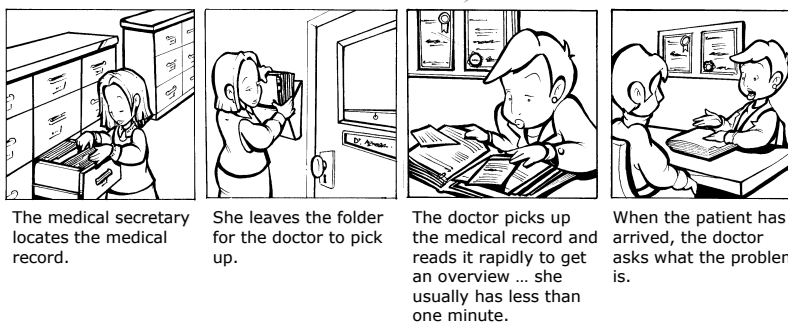
Task Pattern Example (cont'd)

- **Name:** Patient's Record Review
- **Context:** A doctor wants to review a patient's record before talking with in her office (she has few minutes to do this). The record will typically include related information about illnesses, exams and medications. *describes the designer's goal*
- **Problem:** How can we design the interface to provide an overview and at the same time enable comparison and correlation of data, and pattern detection.
- **Example:**



Task Pattern Example (cont'd)

- **Name:** Patient's Record Review
- **Context:** A doctor wants to review a patient's record before in her office (she has few minutes to do this). The record includes related information about illnesses, exams and medications. *an example is used to clarify the task ... this adds a feeling for the task at hand, and PSA uses a storyboard in order to make the example more vivid and easy to understand*
- **Problem:** How can we design the interface to provide a time enable comparison and correlation of data, and pattern detection.
- **Example:**



Task Pattern Example (cont'd)

- **Forces:**

User

- Medicine doctor
- May be computer illiterate
- Female, 30-something

Context

- Doctor knows mostly all of her patients
- Stressful environment
- Lots of information to be analyzed in a very short time
- Doctor may face the patient
- May require real time interpretation of data

forces describe all the factors (sometimes conflicting) that influence design, directly or indirectly

PSA uses detailed forces, specific to the domain but generic within it

the richness provides all the recurring information that would typically be gathered during task and user analysis



Task Pattern Example (cont'd)

Task

- The medical problem has to be solved quickly
- The medical problem has to be solved correctly (errors could be fatal to the patient)
- There is no implicit workflow since the doctor isn't sure of what she's looking for at the beginning
- An overview is needed in order to give the complete patient medical history
- There is a need for pattern detection about illness → medication → exam
- Details needed (information drill down)

- **Interaction Design Solution:**

- Use information visualization to represent information over time
- Provide detail-on-demand for all relevant objects ... make sure that the user doesn't lose spatial context when drilling down
- When structuring and presenting, follow the information visualization mantra: "Overview first, zoom and filter, then detail on demand."
[Shneiderman, 1996]

communicates interaction design considerations for supporting the task



Task Pattern Example (cont'd)

- **Resulting Sub-task Patterns:**

- Overview
- Zoom
- Filter
- Detail-on-demand
- Relate

- **Resulting Structure & Navigation Design Pattern:**

- Time

- **Resulting Screen Design Patterns:**

- Timeline
- Summary

describe the smaller tasks that are part of the complex task described in the pattern

these are generic, making up building blocks for more complex tasks, while at the same time having their own patterns description with forces, related sub-task patterns and structure & navigation design patterns



Task Pattern Example (cont'd)

- **Resulting Sub-task Patterns:**

- Overview
- Zoom
- Filter
- Detail-on-demand
- Relate

- **Resulting Structure & Navigation Design Pattern:**

- Time

- **Resulting Screen Design Patterns:**

- Timeline
- Summary

suggests the overall way(s) to navigating (and thereby implicitly structuring) the data/functions on the level of the complex task being described in the pattern



Task Pattern Example (cont'd)

- **Resulting Sub-task Patterns:**
 - Overview
 - Zoom
 - Filter
 - Detail-on-demand
 - Relate
 - **Resulting Structure & Navigation Design Pattern:**
 - Time
 - **Resulting Screen Design Patterns:**
 - Timeline
 - Summary
- suggest ways of implementing the design solution*



Sub-Task Pattern Example

- **Name:** Overview
- **Examples:** Aggregated data about a population (age, education level, salary, number of children)
- **Context:** The user needs to quickly understand the essence of a data collection
- **Problem:** How do we find the essential qualities of the data, and how should it be presented?
- **Forces:**
 - There are amounts of data
 - The data possesses critical parameters
 - Humans have limitations
 - There are limitations of the graphical device
 - There are many relationships among data
 - These relationships have unequal importance (for example; all relationships are not important at the highest level)
 - Important parameters and relationships are task dependant



Sub-Task Pattern Example (cont'd)

- **Solution:**

The overview must be the basis for interaction with the data (for instance for performing zoom or drilldown actions).

Map the attributes (parameters) to graphical attributes to the following table:

| Attribute | Quantitative | Ordinal | Nominal |
|-------------|--------------|---------|---------|
| Position | + | + | + |
| Size | + | + | + |
| Gray Scale | o | + | - |
| Orientation | o | o | + |
| Color | o | o | + |
| Texture | o | o | + |
| Shape | - | - | + |



Sub-Task Pattern Example (cont'd)

- **Related Sub-task patterns**

- Zoom
- Filter
- Details-on-demand
- Relate
- History

- **Resulting Structure & Navigation Design Patterns**

- Map
- Time

- **Resulting Screen Design Patterns**

- Summary
- Timeline





Hands-on 2

Preparing for User & Task Analysis

45 min.



Structure & Navigation Design Pattern

- describes ways to structure information and/or functions based on navigational styles, in order to support the user's task
- the structure and the navigation are based on the information described in the task patterns (and sub-task patterns)



Structure & Navigation Design Pattern

- so far, we have *recognized* seven S&N patterns based on the work of [Card & al., 1999], [Mok, 1996] and [Shedroff, 1999]:
 - time
 - map
 - step-by-step
 - matrix
 - hierarchy
 - web
 - layers



S&N Design Pattern Example 1

- **Name:** Time
- **Example:** Gantt chart, photo album
- **Context:** The focus is on time related data. We want to see the evolution of a phenomenon over time.
- **Problem:** Selecting a navigational style and designing a structure that shows the evolution of a *phenomenon* over time.
- **Forces:**
 - No implicit/explicit workflow
 - Time is the principal data attribute
 - Users want to perform data analysis and/or exploration
 - Detail-on-demand may be required
- **Solution:** Look for time-related attributes among data objects. Define a *timeline* for each object in order to present the history of events/episodes. Allow user to travel back and forth over time by navigating along the timeline.
- **Related Structure & Navigation Pattern:**
 - Matrix
- **Resulting Design Patterns:**
 - Timeline
 - Small multiples



S&N Design Pattern Example 2

- **Name:** Matrix
- **Examples:** 3-day weather forecast, medical images, sign language book
- **Context:** We want to make correlation and comparison between inter-related data using a set of common attributes.
- **Problem:** Designing a structure model that displays simultaneously the differences or similarities between sets of data.
- **Forces:**
 - Data may be time-related
 - Inter-related data (at least on a set of shared attributes)
 - The user wants to analyze, compare and differentiate in order to make a decision
 - The user wants to analyze objects with different qualities
 - Detail-on-demand may be required
- **Solution:** Look for common attributes. Define a *frame* to display the attributes and repeat this frame for each of the data to be analyzed. Put the emphasis on the attributes, not on the container.
- **Related Structure & Navigation Design Pattern:**
 - Time
- **Resulting Design Pattern:**
 - Small multiples



Screen Design Pattern

- describes screen UI design solutions
- based on the work of [Tidwell, 1999]



Screen Design Pattern Example

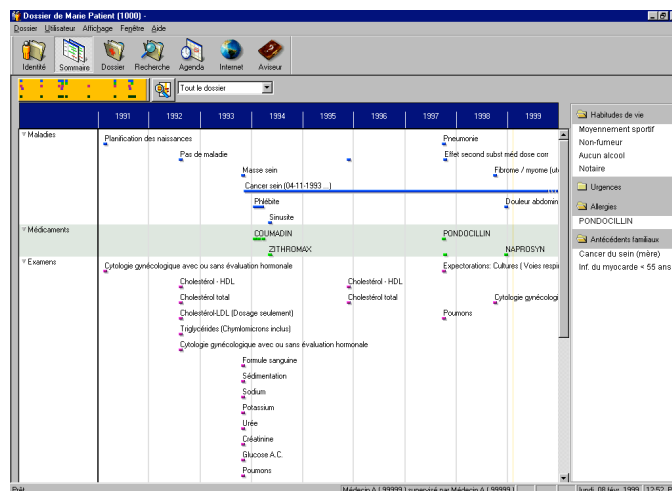
- **Name:** Timeline
- **Examples:** Calendar, *curriculum vitae*, agenda, Gantt chart
- **Context:** There is a need to convey a lot of time-related composite information that may be interrelated.
- **Problem:** How should the information be displayed to the user?
- **Forces:**
 - Data analysis
 - No implicit workflow
 - Overview needed
 - Need for pattern detection
 - Need to correlate data
 - Need to compare data
 - Details needed (information drill down)
 - Information has to be easily interpreted with accuracy
- **Solution:**

Implement timeline into which you group related data vertically, enabling comparison between groups of data, supplying detail on demand, displaying events and episodes on the horizontal axis. Offer zooming, filtering, emphasizing and searching capabilities.



Screen Design Pattern Example (cont'd)

- **Example:**



Part 4

Capturing Patterns



Capturing Patterns

- patterns are **not** invented:
they are **discovered**
- we still have to formally document ...
- ... and this is not an easy thing !



What Makes a Good Pattern?

- it solves a problem !
- it is valid
- non-obvious solution
- it describes relationship
- it adds to human happiness :-)
- it is fit for use



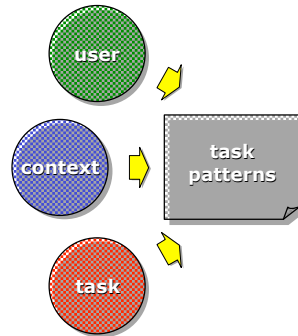
Approach

- we don't think this can be automated!
- we're providing a starting point by pointing out *core characteristics* that we can make predictions for regarding design implications (structure and UI)



Task Pattern Forces

- things to look for while defining the forces of a task pattern



User Forces

- age
- gender
- education
- familiarity with technology
- familiarity with task
- culture
- language
- literacy
- physical impairments
- personal values



Context Forces

- indoors/outdoors, office/home
- fatigue, stress
- interruptions
- time to complete
- error impact
- interactions with others
- security issues
- workspace: private/shared, mobile/stationary, closed (office)/open (cubicle), facing someone



Task Forces

- goal
- trigger (time, visit, phone call, etc.)
- purpose (give info., entertain, educate, guide, sell, analyze)
- complexity
- duration
- frequency
- artifacts used (forms, notes, calendar, etc.)
- workflow
- formal/informal aids (dictionary, procedure manual, colleague)





Hands-on 3

Describing a Task Pattern

90 min.



Part 5

Discussion & Conclusion



Conclusion

- our work on HCI is just beginning ...
- please share with us your experiences using patterns and PSA



References

- ▶ Alexander, C. et al. (1977). *A Pattern Language: Towns, Buildings, Construction*. Oxford University Press, New York, NY.
- Appleton, B. *Patterns and Software: Essential Concepts and Terminology*. <http://www.enteract.com/~bradapp/docs/patterns-intro.html>
- Beyer, H. & Holtzblatt, K. (1998). *Contextual Design: Defining Customer-Centered Systems*. Morgan Kaufmann, San Francisco, CA.
- Borchers, I. *Designing Interactive Music Systems: A Pattern Approach*.
- Borchers, I. (2000). *Pattern Languages for Interaction Design: Building Momentum*. Workshop. CHI 2000, the Hague, Holland.
- Casaday, G. (1998). *Discovering Design Patterns for Interactive Systems*. Workshop. UPA 98, Washington, DC.
- Card, S., Mackinlay, J. & Shneiderman, B. (1999). *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann, San Francisco, CA.
- Cooper, A. (1999). *The Inmates are Running the Asylum*. SAMS.
- ▶ Erickson, T. *The Interaction Design Patterns Page*. http://www.pliant.org/personal/Tom_Erickson/InteractionPatterns.html



References (cont'd)

- Erickson, T. (1997). *Putting It All Together: Pattern Languages for Interaction Design*. Workshop. CHI 97 Conference, Atlanta, GE.
- Erickson, T. (1998). *Interaction Pattern Languages: A Lingua Franca for Interaction Design?* UPA 98 Conference, Washington, DC.
- Fowler, M. (1997). *Analysis Patterns: Reusable Object Models*. Addison Wesley, Menlo Park, CA.
- Granlund, Å. & Lafrenière, D. (1999). *A Pattern-Supported Approach to User Interface Design*. Workshop. UPA 99, Scottsdale, AZ.
- Hackos, J. & Redish, J. (1998). *User and Task Analysis for Interface Design*. John Wiley & Sons, New York, NY.
- Lafrenière, D. (1998). *From Entity-Relationship Diagram and Task Analysis Diagram to User Interface Architecture*. Workshop Presentation. UPA 98, Washington DC.
- Lea, D. *Patterns-Discussion FAQ*. <http://g.oswego.edu/dl/pd-FAQ/pd-FAQ.html>
- Loureiro, K. & Plummer, D. (1999). *AD Patterns: Beyond Objects and Components*. Research Note # COM-08-0111, Gartner Group.
- Mok, C. (1996). *Designing Business*. Adobe Press, San Jose, CA.



References (cont'd)

- Robertson, S. Requirements Patterns via Events/Use Cases. http://www.atlsysguild.com/GuildSite/SQR/Requirements_Patterns.html
- Robertson, S. Reusing the Products of Analysis. <http://www.atlsysguild.com/GuildSite/SQR/reusingAnalysis.html>
- Shedroff, N. (1999). *Information Interaction Design: A Unified Field Theory of Design in Information Design*. MIT Press, Cambridge, MA.
- Shneiderman, B (1996). The eyes have it: a task by data type taxonomy for information visualizations, *Proceedings of 1996 IEEE Visual Languages*, 336-343.
- Tidwell, J. (1999). *Common Ground: A Pattern Language for Human-Computer Interface Design*. http://www.mit.edu/~jtidwell/interaction_patterns.html
- Tufte, E. (1983). *The Visual Display of Quantitative Information*. Graphic Press. Cheshire, CO.
- Tufte, E. (1990). *Envisioning Information*. Graphic Press. Cheshire, CO.
- Tufte, E. (1997). *Visual Explanations*. Graphic Press. Cheshire, CO.
- Wurman, R.S. (1996). *Information Anxiety*. Doubleday, New York, NY.
- Wurman, R.S. (1996). *Information Architects*. Graphis Press, Zurich, Switzerland.

