

# Conceptual Architecture

John Reekie  
University of Technology, Sydney

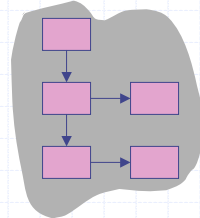
Contributors  
Lian Loke, University of Technology Sydney

Terms of Use: Creative Commons Attribution-ShareAlike 2.5  
<http://creativecommons.org/licenses/by-sa/2.5/>

## Recall architectural views

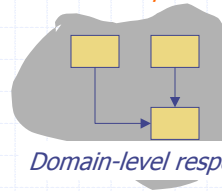
A view expresses a particular cross-section of architectural concerns

*Implementation architecture*



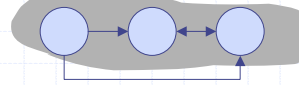
*Build-time structure*

*Conceptual architecture*



*Domain-level responsibilities*

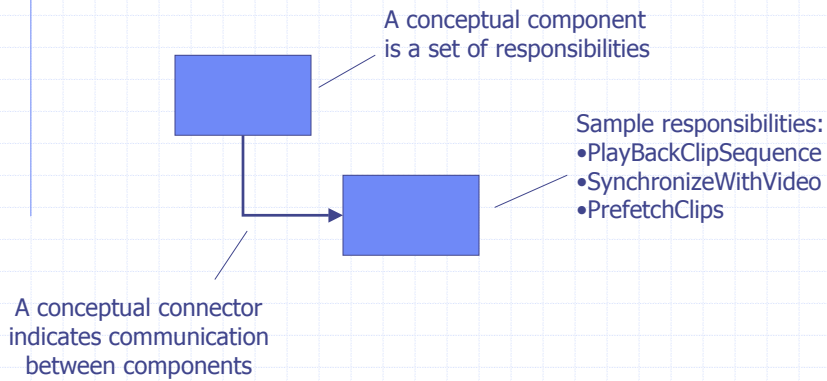
*Execution architecture*



*Run-time structure*

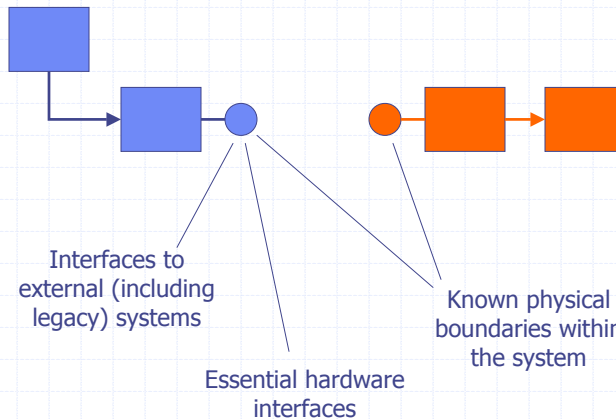
In this lecture, we focus on the  
*conceptual architecture*

## Elements of conceptual architecture



The **conceptual architecture** structures the system in terms of its domain-level responsibilities

## External interfaces



Note: A stakeholder is **not** an external system!

## Solving a complex problem



*You have to start somewhere...*

### 1. Obtain a system narrative

Custom Shooz plan to advertise using conventional means, but want the website to be a location where customers can find out about their custom range, get the measurement kit, and customize and order shoes. They also want the site to interface to their accounting system.

The President of Custom Shooz, Funk O. Sole, explains:  
"So, what we did was develop a little measurement kit that we send out to folks and they have to send it back. We've improved it over the last couple of years so that it's almost foolproof. Once we have the customer's measurement kit in, we can produce almost any shoe from our range - all the custom stuff, like stitched-on patterns, dye colours and finishes, laces and buckles, can be done without them ever being within a thousand miles of our store!"

(Extract from system narrative in Workbook)

## 2. Identify key concepts

Custom Shooz plan to **advertise** using conventional means, but want the **website** to be a location where **customers** can find out about their **custom range**, get the **measurement kit**, and **customize and order shoes**. They also want the site to **interface to their accounting system**.

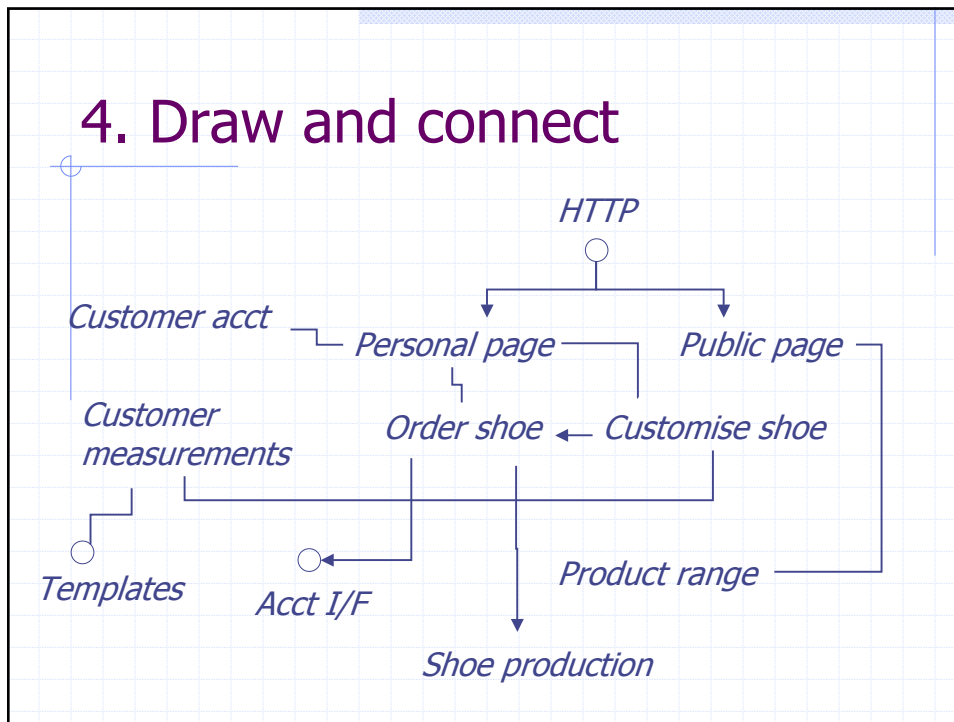
The President of Custom Shooz, Funk O. Sole, explains:

"So, what we did was develop a little **measurement kit** that we send out to folks and they have to send it back. We've improved it over the last couple of years so that it's almost foolproof. Once we have the customer's measurement kit in, we can **produce almost any shoe** from our **range** - all the custom stuff, like **stitched-on patterns, dye colours and finishes, laces and buckles**, can be done without them ever being within a thousand miles of our store!"

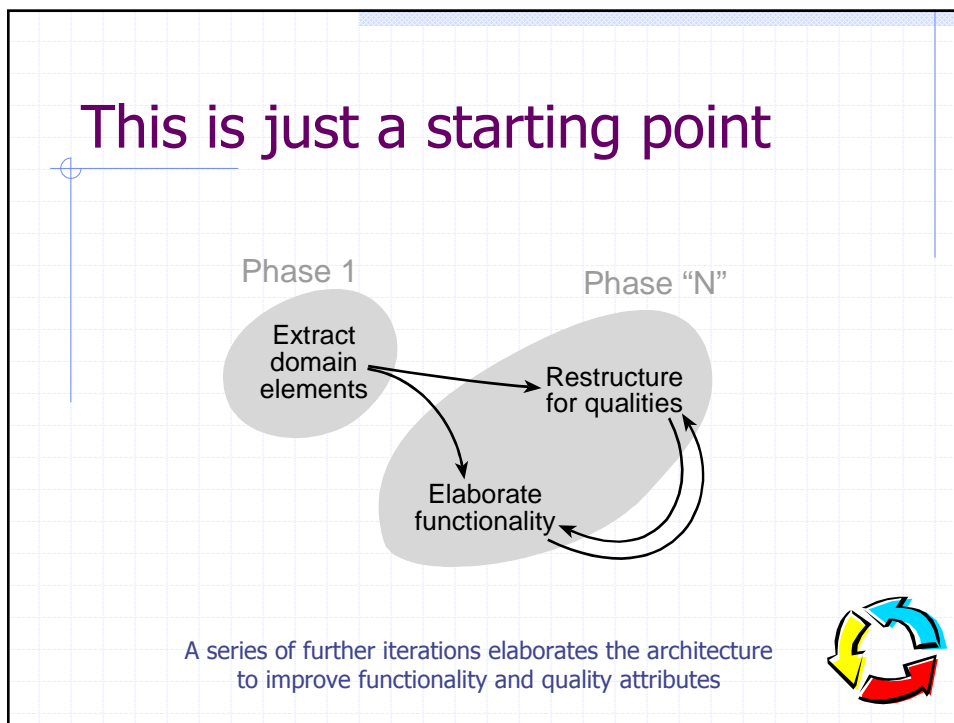
## 3. Refine to components

Advertise - abstract concept ⇒ X  
Website - implementation ⇒ ?  
Customers - stakeholder ⇒ **Customer account + Personalised page**  
Custom range ⇒ **Product range**  
Measurement kit ⇒ **Customer measurements**  
**Customise shoe** ✓  
**Order shoe** ✓  
Accounting I/F - external system ⇒ **Acct I/F**  
Produce shoe - function/external system ⇒ **Shoe production**  
Patterns and finishes - part of Customise shoe

## 4. Draw and connect

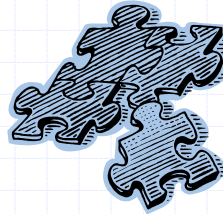


## This is just a starting point



## Refine the architecture

- ◆ Add or split components
- ◆ Clarify responsibilities
- ◆ Identify stereotypes
- ◆ Create data models
- ◆ Explore behavior

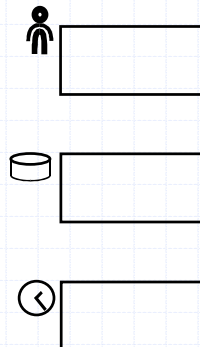


A component is a set of *related* responsibilities. So, split a component if responsibilities are *not* related ...



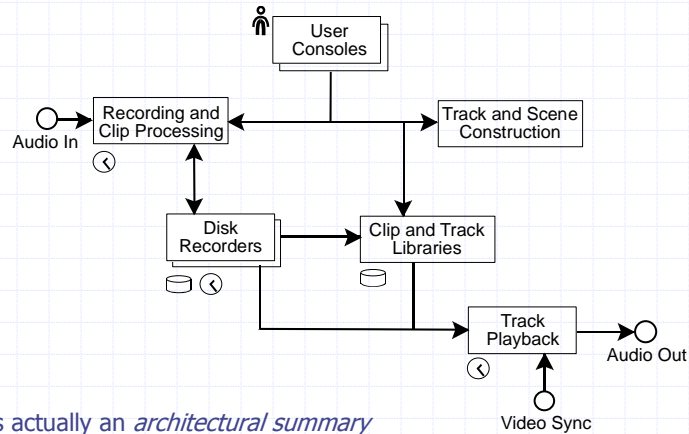
## Conceptual stereotypes

- ◆ Does a component have special types of responsibilities?
  - User presentation
  - Persistent storage
  - Realtime response



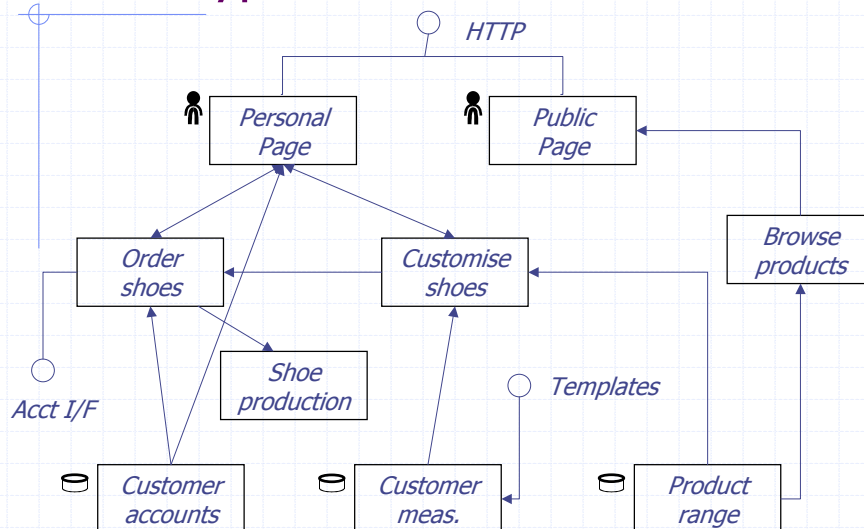
A **stereotype** indicates that a component (or in UML, a class) has certain properties or attributes.

## A stereotype example



This is actually an *architectural summary diagram*. It is useful for providing an at-a-glance overview of a complex system.

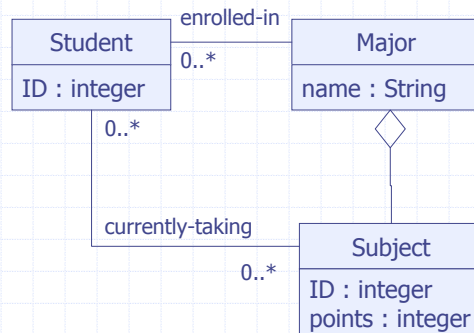
## Custom Shooz architecture with stereotypes



## Data models

◆ A data model captures the *essential* structure of data

- Data along connectors
- Persistent data



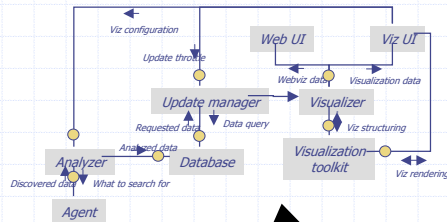
## What is behaviour



A system has function, structure *and* behaviour

- ◆ Behaviour is the set of actions that the system performs
- ◆ Behaviour can be explored through:
  - Role-play
  - Use Case maps
  - Sequence diagrams

## How can we explore deeper?



The system must have some behavior in response to activity in the usage narratives

One, two! One, two! And through and through  
The vorpal blade went snicker-snack!  
He left it dead, and with its head  
He went galumphing back..

He took his vorpal sword in hand:  
Long time the manxome foe he sought  
So rested he by the Tumtum tree,  
And stood awhile in thought.

And as in uffish thought he stood,  
The Jabberwock, with eyes of flame,  
Came whiffing through the tulgey wood,  
And burbled as it came



## Extract events from narratives

Julie is interested in correlating sightings of Perameles Nasuta in the Northern beaches area of Sydney with bushfire patterns. She *brings up tracking data* for the last five years and proceeds to *sort the data*, and then *export* it into a form that it can be used by a statistical analysis package



*RequestHistoricalTrackingData*

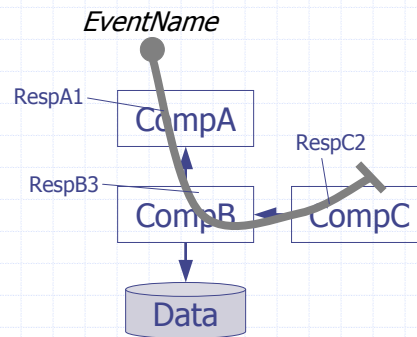
*SortHistoricalTrackingData*

*ExportHistoricalTrackingData*

## Events trigger use-case maps

◆ Use-case maps allow us to visualise a path of action through a system

- A trace shows the sequence of activities
- Activity is triggered by an event
- Each time the trace crosses a component, it exercises a responsibility



Use-case maps facilitate understanding of macroscopic behaviour

## Conceptual arch. example

## Summary

- ◆ The conceptual architecture focuses on conceptual components
  - Domain-level responsibilities
  - Links/refines requirements
  - Overall/initial system structure
- ◆ Includes data models
  - Sometimes a separate *information view* is used
- ◆ It is *essential* to explore behavior
  - Identifies missing responsibilities
  - Ties the architecture back to narratives/requirements

## That's all, folks!

- ◆ Questions or comments?

