“People react positively when things are clear and understandable.”

Dieter Rams
THE BEST APPLICATIONS
REACT TO THE USER'S WORKFLOW
NOT
THE OTHER WAY AROUND
SWT, OSGI & MODULARITY IN MEDICAL IMAGING

Juan Narvaez
Software Architect

Cerner

“As far as the customer is concerned, the interface is the product.”
KNOW

USABILITY

RELIABILITY

SIMPLICITY
Know your solution's target audience
KNOW
USABILITY
RELIABILITY
SIMPLICITY
Make your audience want to use your solution

Know your solution's target audience

Make your solution fault tolerant

Make your solution

tolerant
KNOW USABILITY RELIABILITY SIMPLICITY
Make your solution fault tolerant
KNOW
USABILITY
RELIABILITY

SIMPLICITY

START WITH THE
VISUALS
OF YOUR SOLUTION
Make your solution fault tolerant. Make your solution intuitive, for both consumers and developers.
START WITH THE VISUALS OF YOUR SOLUTION
WHY?
“As far as the customer is concerned, the interface is the product.”

Jef Raskin
LONG HOURS

60+ HRS/WEEK

16,000+ PROCEDURES PER USER PER YEAR

(staring at the same application)
USE

SWT & OSGI

FOR MODULAR PURPOSE BUILT SOLUTIONS
HOW?
“Before software can be reusable it first has to be usable.”

Ralph Johnson
MODULAR COMPONENTS

Create **reusable** SWT widgets

Tailors the look and feel to the purpose of the solution

Employ OSGi **services** when possible

Allows for specific assemblies and rapid development
Native widgets provide a natural look and feel

Provides the ability to build a professional looking solution quickly

Integrates with JFace for easy MVC development

Limiting for low-light workflows in imaging

Provide analogous widget set that is skinnable

Allows the solution to be tailored to any visual requirements

Focus on the low light workflow
Native widgets provide a natural look and feel

Provides the ability to build a professional looking solution quickly

Integrates with JFace for easy MVC development

Limiting for low-light workflows in imaging

Provide analogous widget set that is skinnable

Allows the solution to be tailored to any visual requirements

Focus on the low light workflow
Native widgets provide a natural look and feel

Provides the ability to build a professional looking solution quickly

**Integrates with JFace for easy MVC development**

Limiting for low-light workflows in imaging

Provide analogous widget set that is skinnable

Allows the solution to be tailored to any visual requirements

Focus on the low light workflow
Native widgets provide a natural look and feel

Provides the ability to build a professional looking solution quickly

Integrates with JFace for easy MVC development

Limiting for low-light workflows in imaging

Provide analogous widget set that is skinnable

Allows the solution to be tailored to any visual requirements

Focus on the low light workflow
Native widgets provide a natural look and feel

Provides the ability to build a professional looking solution quickly

Integrates with JFace for easy MVC development

Limiting for low-light workflows in imaging

Provide analogous widget set that is skinnable

Allows the solution to be tailored to any visual requirements

Focus on the low light workflow
Native widgets provide a natural look and feel

Provides the ability to build a professional looking solution quickly

Integrates with JFace for easy MVC development

Limiting for low-light workflows in imaging

Provide analogous widget set that is skinnable

Allows the solution to be tailored to any visual requirements

Focus on the low light workflow
Native widgets provide a natural look and feel

Provides the ability to build a professional looking solution quickly

Integrates with JFace for easy MVC development

Limiting for low-light workflows in imaging

Provide analogous widget set that is skinnable

Allows the solution to be tailored to any visual requirements

Focus on the low light workflow
MODULAR COMPONENTS

Create **reusable** SWT widgets

Tailors the look and feel to the purpose of the solution

Employ OSGi **services** when possible

Allows for specific assemblies **and** rapid development
MODULAR COMPONENTS

Create reusable SWT widgets

Tailors the look and feel to the purpose of the solution

Employ OSGi services when possible

Allows for specific assemblies and rapid development
Inject different implementations at runtime

Use same services and interfaces on both clients and servers

Different assemblies just contain different service implementations based on the use case
Inject different implementations at runtime

Use same services and interfaces on both clients and servers

Different assemblies just contain different service implementations based on the use case
Inject different implementations at runtime

Use same services and interfaces on both clients and servers

Different assemblies just contain different service implementations based on the use case
MODULAR COMPONENTS

Create **reusable** SWT widgets

Tailors the look and feel to the purpose of the solution

Employ OSGi **services** when possible

Allows for specific assemblies **and** rapid development
WHAT?
“The behavior you’re seeing is the behavior you’ve designed for (whether intentional or not).”

Joshua Porter
RETHINK
THE LAYOUT
WHAT THE USER CAN DO
HOW IT RESPONDS
HOW THE SOLUTION IS
DELIVERED
CREATE THE
BASICS
SKINNABLE WIDGETSET

![Widget Description]

<table>
<thead>
<tr>
<th>Name</th>
<th>comp</th>
</tr>
</thead>
<tbody>
<tr>
<td>Patient ID</td>
<td>Date of Birth</td>
</tr>
<tr>
<td>Record 1</td>
<td>Record 2</td>
</tr>
<tr>
<td>Record 3</td>
<td>Record 4</td>
</tr>
<tr>
<td>Record 5</td>
<td>Record 6</td>
</tr>
<tr>
<td>Record 7</td>
<td>Record 8</td>
</tr>
<tr>
<td>Record 9</td>
<td>Record 10</td>
</tr>
</tbody>
</table>

![Table Data]

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Date/Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Record 1</td>
<td>Description 1</td>
<td>Date/Time 1</td>
</tr>
<tr>
<td>Record 2</td>
<td>Description 2</td>
<td>Date/Time 2</td>
</tr>
<tr>
<td>Record 3</td>
<td>Description 3</td>
<td>Date/Time 3</td>
</tr>
<tr>
<td>Record 4</td>
<td>Description 4</td>
<td>Date/Time 4</td>
</tr>
<tr>
<td>Record 5</td>
<td>Description 5</td>
<td>Date/Time 5</td>
</tr>
<tr>
<td>Record 6</td>
<td>Description 6</td>
<td>Date/Time 6</td>
</tr>
<tr>
<td>Record 7</td>
<td>Description 7</td>
<td>Date/Time 7</td>
</tr>
<tr>
<td>Record 8</td>
<td>Description 8</td>
<td>Date/Time 8</td>
</tr>
<tr>
<td>Record 9</td>
<td>Description 9</td>
<td>Date/Time 9</td>
</tr>
<tr>
<td>Record 10</td>
<td>Description 10</td>
<td>Date/Time 10</td>
</tr>
</tbody>
</table>
SKINNABLE WIDGETSET
SKINNABLE WIDGETSET
SKINNABLE WIDGETSET
USE THE WORKBENCH TO HANDLE THE LAYOUT
USE THE WIDGETS TO PROVIDE THE LOOK

PUT IT

TOGETHER
GREAT SOLUTION

NOW

WHAT?
DIAGNOSTIC IMAGING REQUIRES HEAVY DUTY HARDWARE

($10,000+ Monitors, and multiple at that!)

CLINICIANS, NURSES, SURGEONS, ETC. WANT THE SAME TOOLS AND CAPABILITIES

(Without the access to the hardware)

(and they don't want to install anything)
Diagnostic imaging requires heavy duty hardware
($10,000+ monitors, and multiple at that!)

Clinicians, nurses, surgeons, etc. want the same tools and capabilities

(Without the access to the hardware)

(and they don't want to install anything)
NEED A WAY TO DELIVER
THE SAME SOLUTION TO THE ENTERPRISE
CITRIX?
VMWARE?
VMWARE?

REQUIRE PLUGINS
NETWORK INTENSIVE FOR IMAGING

(Up to 3MB per second!)

USE THE MODULES WE ALREADY HAVE
BUILD ON WHAT THE OTHER GUYS DO
USE THE MODULES WE ALREADY HAVE

BUILD ON WHAT THE OTHER GUYS DO

(REMOTE DESKTOP SERVICES)
VM WARE?

REQUIRE PLUGINS
NETWORK INTENSIVE FOR IMAGING

USE THE MODULES WE ALREADY HAVE
BUILD ON WHAT THE OTHER GUYS DO
(REMOTE DESKTOP SERVICES)

Diagram:
- Equinox Runtime Jetty
  - HID Input Services
  - Rendering Services
- RDS Services
  - Application Session Services
  - Use RDS to create application sessions
  - Use web services to start/stop sessions, lookup running sessions, send information to running application instances
  - Connect to running instance or start new RDS session
  - Serve up renderings of the application session to a client browser
SO WHAT DOES IT LOOK LIKE?
NOW WE CAN RUN ON
ANYTHING
WITH A BROWSER!
WHAT'S NEXT?
THIRD PARTIES CAN CONTRIBUTE MODULES FOR SPECIALIZED FUNCTIONALITY

(WE'VE GOT AN SDK!)
SO WHAT DO YOU THINK?
ENGINEERING HEALTH at CERNER

JUAN NARVAEZ on GitHub

CERNER.COM

Written in JavaScript/HTML5/CSS3 with the help of impress.js

“Design with the end in mind.”