CS5386
SOFTWARE DESIGN & ARCHITECTURE

Lecture 6:
ARCHITECTURAL VIEWS
ALLOCATION STYLES

Outline for Today

- Schedule for rest of semester
- Allocation Styles
- Group work C&C View
### Schedule for Remainder of the Semester

<table>
<thead>
<tr>
<th>Date</th>
<th>Class Activity</th>
<th>Team Deliverable(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>03/24</td>
<td>Allocation View</td>
<td>C&amp;C view 1st Draft</td>
</tr>
<tr>
<td>03/31</td>
<td>No class</td>
<td></td>
</tr>
<tr>
<td>04/07</td>
<td>Documenting behavior, interfaces and context</td>
<td>Allocation View Draft</td>
</tr>
<tr>
<td>04/14</td>
<td>AADL</td>
<td>Complete Architecture Document (First Draft)</td>
</tr>
<tr>
<td>04/21</td>
<td>AADL</td>
<td>Complete Architecture Document (Final Draft)</td>
</tr>
<tr>
<td>04/28</td>
<td>In class group work</td>
<td>C&amp;C in AADL First Draft</td>
</tr>
<tr>
<td>05/05</td>
<td>Team Presentations</td>
<td>C&amp;C in AADL Final Draft</td>
</tr>
<tr>
<td>05/12</td>
<td>Final Exam</td>
<td></td>
</tr>
</tbody>
</table>

### Objectives

- Identify a number of allocation styles
- Describe how to document views in each of the styles
Allocation Styles

- **Elements:**
  - software elements (as defined in module or C&C styles)
  - environment elements

- **Relations:** “allocated to”

- **Properties:**
  - requires and provides properties, appropriate to each style

Allocation Styles: Software Elements and Environment Elements

- Deployment style
- Install style
- Work assignment style
Deployment Style - 1

- **Overview:**
  - allocates software elements to processing and communication nodes

- **Elements:**
  - software element—for example, processes or services from a C&C view
  - environment element—computing hardware

- **Relations:**
  - “allocated to”—physical elements on which software resides
  - “migrates to,” “copy migrates to,” and/or “execution migrates to” with dynamic allocation.

Deployment Style - 2

- **Properties:**
  - Properties include those necessary to calculate and achieve performance and availability.
  - Significant features *provided* by hardware elements: for example, CPU speed, number of processors, memory size, disk size, bandwidth, fault tolerance.
  - Features *required* by software elements: for example, memory consumption, disk space needed, response time of components, availability required.

- **Constraints:** unrestricted

- **What it’s for:** analysis of
  - performance
  - bandwidth utilization
  - availability
  - security
  - purchasing options for hardware
Deployment Style: Example


Deployment Style: UML Example
Install Style - 1

- **Overview:**
  - describes the mapping of components in the software architecture to a file system in the production environment

- **Elements:**
  - software element—a C&C component
  - environment element—a configuration item, such as a file or a folder

- **Relations:**
  - allocated-to —A component is allocated to a configuration item.
  - containment—One configuration item is contained in another.

Install Style - 2

- **Properties:**
  - **Required** properties of a software element: include requirements on the production environments, such as a requirement to support Java or a database, or specific permissions on the file system.

  - **Provided** properties of an environment element: include indications of the characteristics provided by the production environments.
Install Style - 3

- **Constraints:**
  - Files and folders are organized in a tree structure, following an is-contained-in relation.

- **What it’s for:**
  - creating build-and-deploy procedures
  - navigating through a large number of files and folders that constitute the installed system, to locate specific files that require attention (such as a log file or configuration file)
  - selecting and configuring files to package a specific version of a software product line
  - updating and configuring files of multiple installed versions of the same system
  - identifying the purpose or contents of a missing or damaged file that is causing a problem in production
  - designing and implementing an “automatic updates” feature

---

Install Style: Example

![Diagram](image)

Install view from Duke’s Bank example, in informal notation.

Install Style: Example 2

Overview:
allocates software elements to organizational work units

Elements:
- software element—C&C or Modules???
- environment element—organizational unit; for example, a person, team, or subcontractor

Relations:
“allocated to”—software element allocated to a development team

Properties:
- requires property for software element: skills set and capacity
- provides property of environment elements: skills set and capacity
Work Assignment Style - 2

- **Constraints:**
  - Unrestricted in general
  - Different in practice:
    - usually one module is allocated to one organizational unit (can you think of a case where one module is assigned to multiple teams?)
    - an organizational unit can produce more than one module

- **What it’s for:**
  - showing the major units of software necessary for a system and who will produce (i.e., develop, test, and integrate) them
  - planning/managing team resource allocations
  - explaining the structure of a project
  - assigning responsibilities for builds
  - early on, helping to write the request for proposals

Work Assignment Style - Example

<table>
<thead>
<tr>
<th>ECS Element (Module)</th>
<th>Subsystem</th>
<th>Organizational unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Science Data Processing Segment (SDPS)</td>
<td>Client</td>
<td>Science team</td>
</tr>
<tr>
<td></td>
<td>Interoperability</td>
<td>Prime contractor team 1</td>
</tr>
<tr>
<td></td>
<td>Ingest</td>
<td>Prime contractor team 2</td>
</tr>
<tr>
<td></td>
<td>Data Management</td>
<td>Data team</td>
</tr>
<tr>
<td></td>
<td>Data Processing</td>
<td>Data team</td>
</tr>
<tr>
<td></td>
<td>Data Server</td>
<td>Data team</td>
</tr>
<tr>
<td></td>
<td>Planning</td>
<td>Orbital vehicle team</td>
</tr>
<tr>
<td>Flight Operations Segment (FOS)</td>
<td>Planning and Scheduling</td>
<td>Orbital vehicle team</td>
</tr>
<tr>
<td></td>
<td>Data Management</td>
<td>Database team</td>
</tr>
<tr>
<td></td>
<td>User Interface</td>
<td>User interface team</td>
</tr>
</tbody>
</table>

Do we want architects handing out work assignments to organizational teams?
Work Assignment Style: Specializations

- **Platform style:**
  - In a software product line development, one site is tasked with developing reusable core assets of the product line, and other sites develop applications that use the core assets.

- **Competence-center style:**
  - Work is allocated to sites depending on the technical or domain expertise located at a site. For example, user-interface design is done at a site where usability engineering experts are located.

- **Open-source style:**
  - Many independent contributors develop the software product in accordance with a technical integration strategy. Centralized control is minimal, except when an independent contributor integrates his code into the product line.

---

Schedule for Remainder of the Semester

<table>
<thead>
<tr>
<th>Date</th>
<th>Class Activity</th>
<th>Team Deliverable(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>03/24</td>
<td>Allocation View</td>
<td>C&amp;C view 1st Draft</td>
</tr>
<tr>
<td>03/31</td>
<td>No class</td>
<td></td>
</tr>
<tr>
<td>04/07</td>
<td>Documenting behavior, interfaces and context</td>
<td>Allocation View Draft</td>
</tr>
<tr>
<td>04/14</td>
<td>AADL</td>
<td>Complete Architecture Document (First Draft)</td>
</tr>
<tr>
<td>04/21</td>
<td>AADL</td>
<td>Complete Architecture Document (Final Draft)</td>
</tr>
<tr>
<td>04/28</td>
<td>In class group work</td>
<td>C&amp;C in AADL First Draft</td>
</tr>
<tr>
<td>05/05</td>
<td>Team Presentations</td>
<td>C&amp;C in AADL Final Draft</td>
</tr>
<tr>
<td>05/12</td>
<td>Final Exam</td>
<td></td>
</tr>
</tbody>
</table>