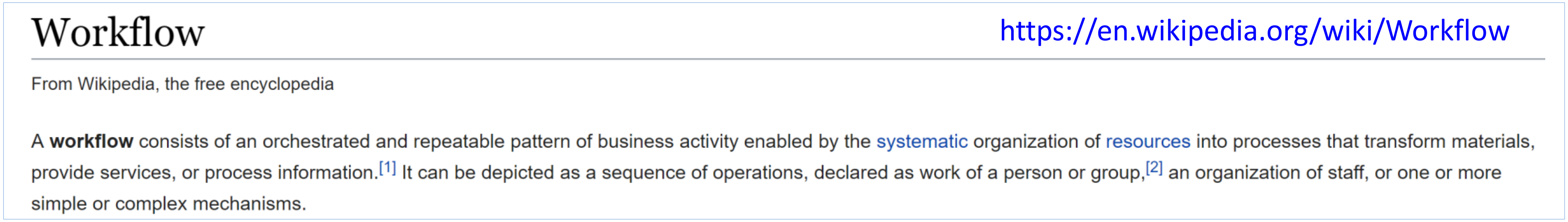
Loosely-Coupled/Tightly-Constrained (LCTC) Workflows for Inter-organizational, Compliance-Governed, Collaborative Computing for Multi-Tenant Cloud

Marc Edwards (SCI), Rhett Davis (NCSU), Paul Rad (UTSA)

Taxonomy of Loosely Couple Tightly Constrained Workflows



Workflow

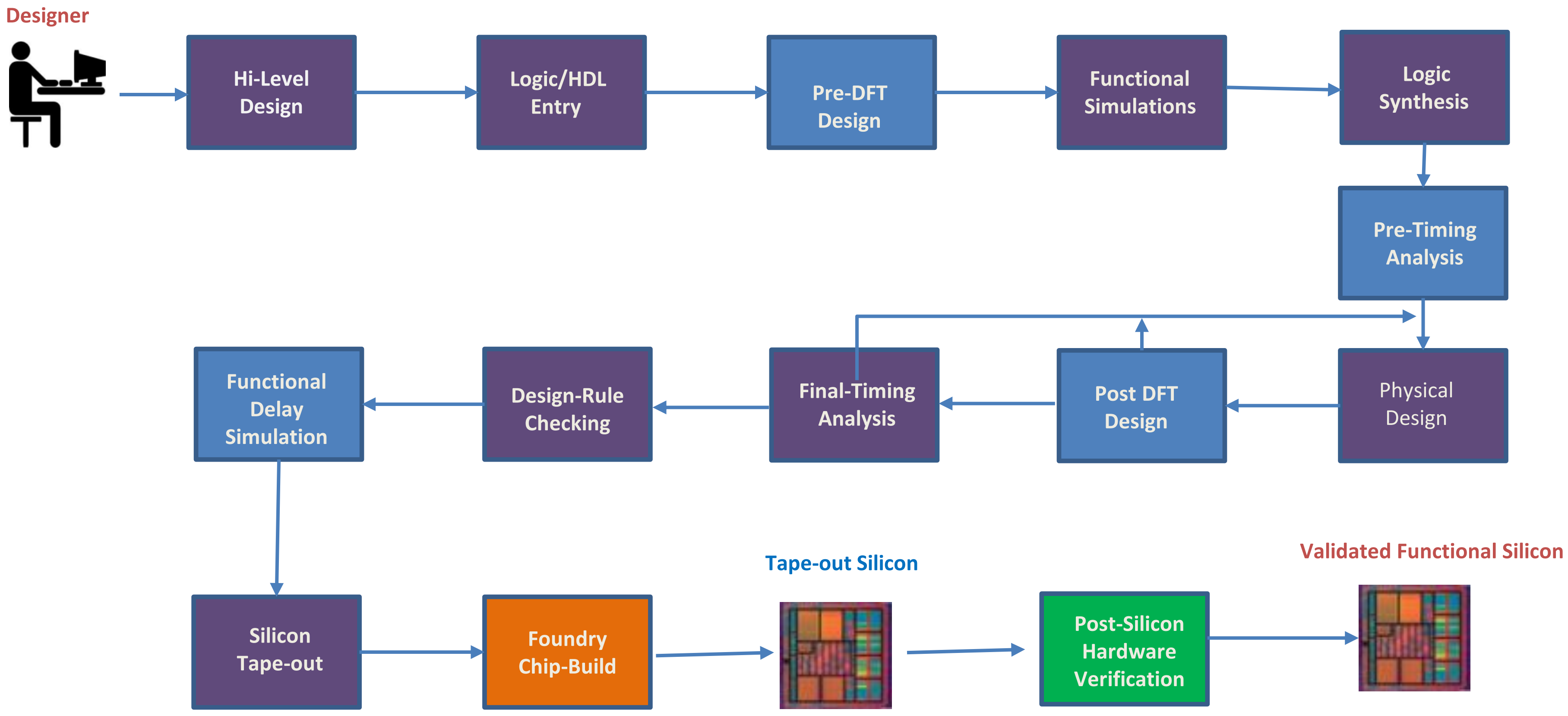
https://en.wikipedia.org/wiki/Workflow

From Wikipedia, the free encyclopedia

A **workflow** consists of an orchestrated and repeatable pattern of business activity enabled by the systematic organization of resources into processes that transform materials, provide services, or process information.[1] It can be depicted as a sequence of operations, declared as work of a person or group,[2] an organization of staff, or one or more simple or complex mechanisms.

- We seek a computational framework for (1) IP DRM, (2) COMPUTATIONAL COMPLIANCE, (3) SECURE COLLABORATION (IP SHARING)
- ***Engineering Workflows (EW)*** dominate engineering design practices, e.g. semiconductor, FEA, engine analysis, tire design are well defined EWs.
- We distinguish ***Engineering Workflows*** from ***Scientific Workflows (SW)***
  - SWs are “tightly-coupled” steps, EWs are “loosely-coupled” “phases”
- ***Engineering Workflows*** are classified as (1) ***Design To Release Manufacturing*** (DTRM) workflows, i.e. the computing activities that lead UP TO the point of releasing the product in contrast to (2) ***Manufacturing To Deployment*** (MTD) workflows.
- *“sequence of operations”* ➔ EW “sequences” can be described as ***PHASES*** of a workflow.
- EACH ***PHASE*** of an workflow represents a *computational transformation* for an INPUT and OUTPUT dataset(s), i.e. hyper-complex function.
  - LINKED COLLECTION of PHASES constitute the EW

Semiconductor Design Workflow Example



Designer

Hi-Level Design

Logic/HDL Entry

Pre-DFT Design

Functional Simulations

Logic Synthesis

Pre-Timing Analysis

Physical Design

Post DFT Design

Final-Timing Analysis

Design-Rule Checking

Functional Delay Simulation

Silicon Tape-out

Foundry Chip-Build

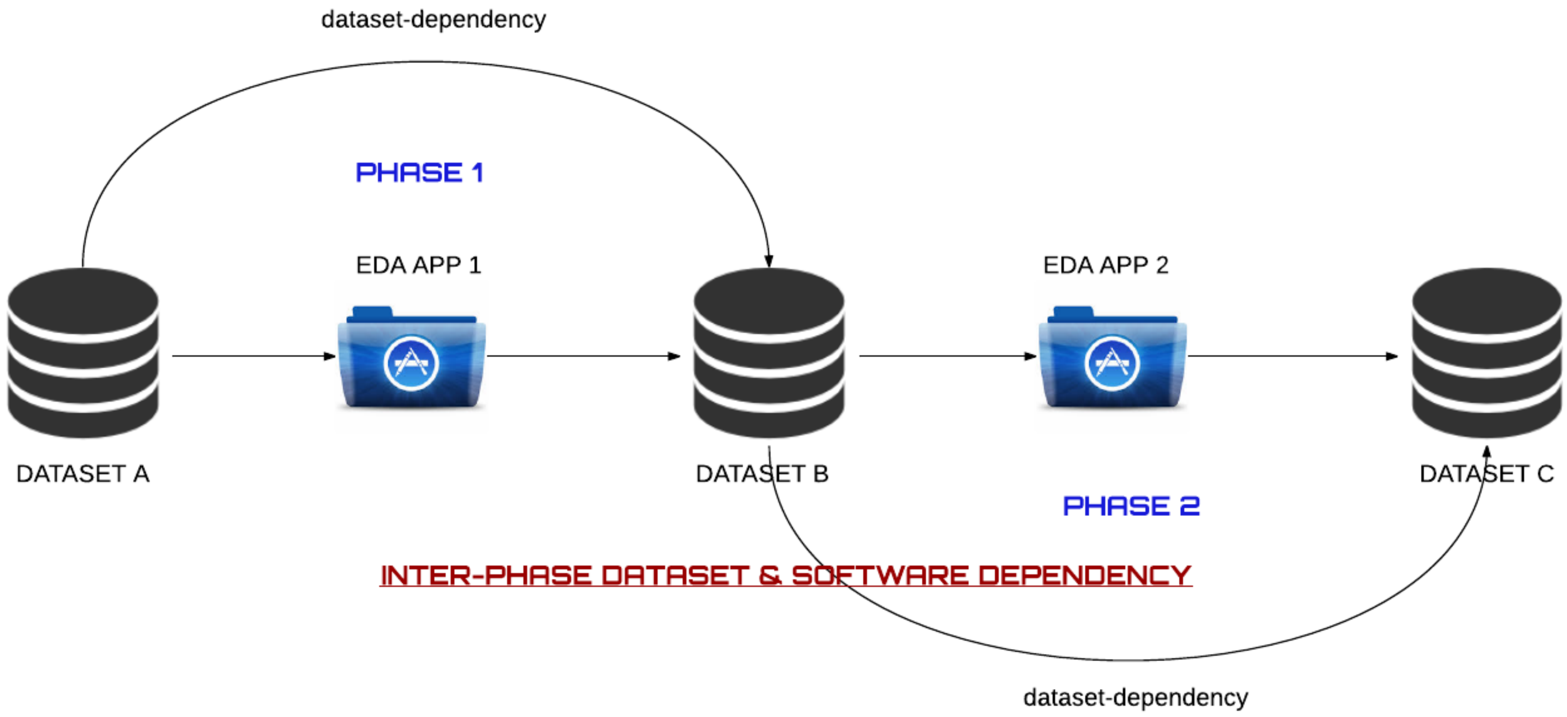
Tape-out Silicon

Post-Silicon Hardware Verification

Validated Functional Silicon

Foundry
Build

Workflow Inter-phase Dependencies



Silicon Cloud Overview

- ❑ EW ***Intellectual Property*** (IP) is represented as INPUT & OUTPUT DATASETS
  - We seek to track the *"transformational provenance"* of an IP's progression through the phases of a workflow, i.e. the *«execution provenance»* of the IP
  - *"Execution Provenance"* of IP exemplifies the ***DRM*** component of the IP.

- ❑ A key and distinguishing characteristic of DTRM EWs from SWs…
  - The INTER-dependencies ("linked") of the *"transformational provenance"* of IP BETWEEN PHASES that are **LOOSELY COUPLED**, but…
    - **"Transformational Provenance" TIGHTLY CONSTRAINS** the IP within the DTRM workflow

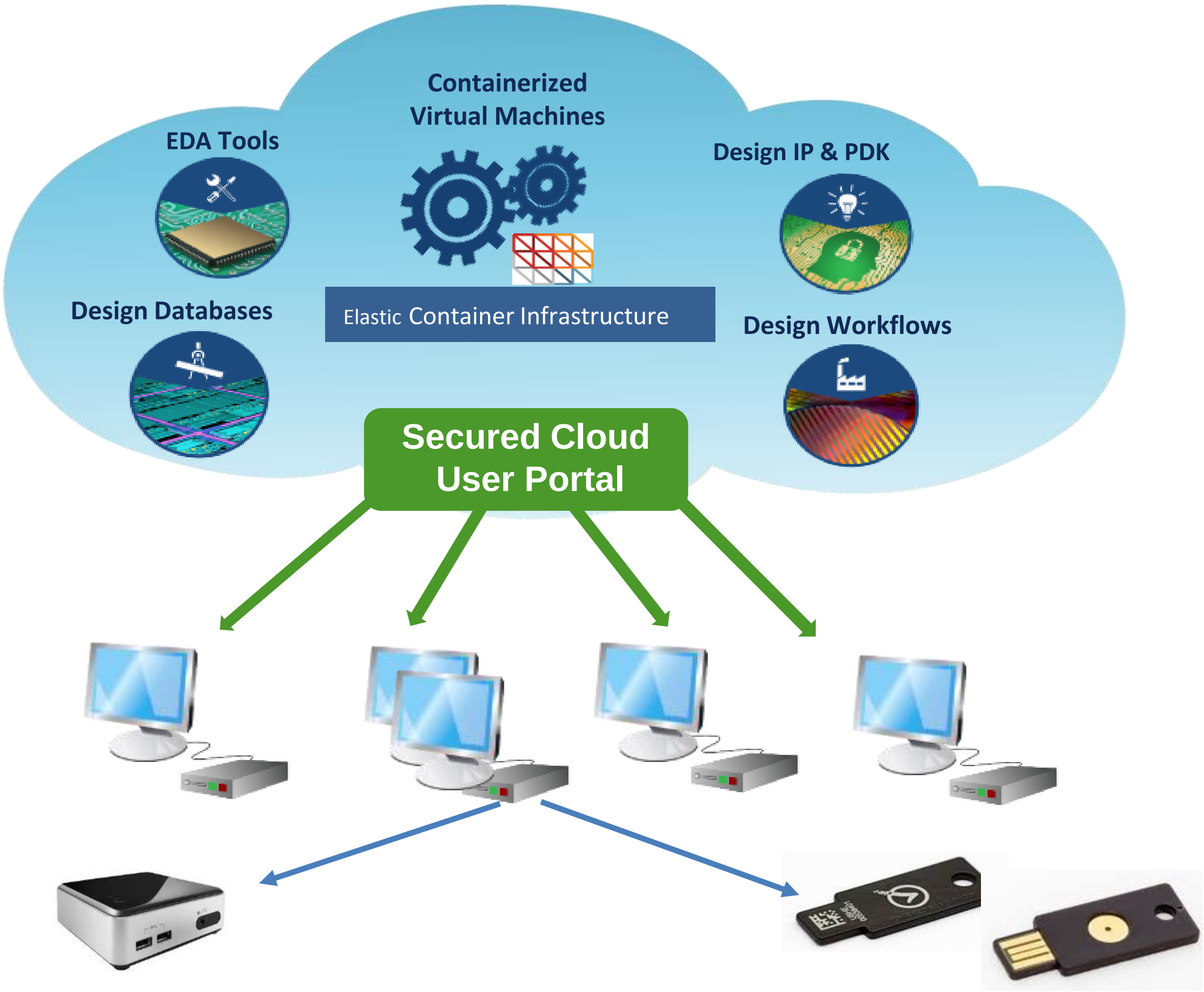
- ❑ ***"TIGHTLY CONSTRAINED"*** component DEFINES the ***COMPLIANCE*** aspect of an ***Loosely-Coupled-Tightly-Constrained*** (LCTC) workflow

- ❑ The multi-phase aspect of the workflow enables a *IP PHASE EXTRACTION/INSERTION POINT*
  - Exposes a SECURITY HOLE within the workflow
  - IP datasets can be "ejected" from the workflow
  - CON ➔ Destroys the IP security within the workflow
  - PRO ➔ Provide a mechanism for DRM through provenance record
  - EW execution often requires multi-person/roles
  - Demands a secure collaboration schema as EW phases are assigned through roles

Silicon Cloud I2 Infrastructure Overview

Private & Secured Cloud

Cloud Access Thin Clients



EDA Tools

Containerized Virtual Machines

Design IP & PDK

Design Databases

Elastic Container Infrastructure

Design Workflows

Secured Cloud User Portal

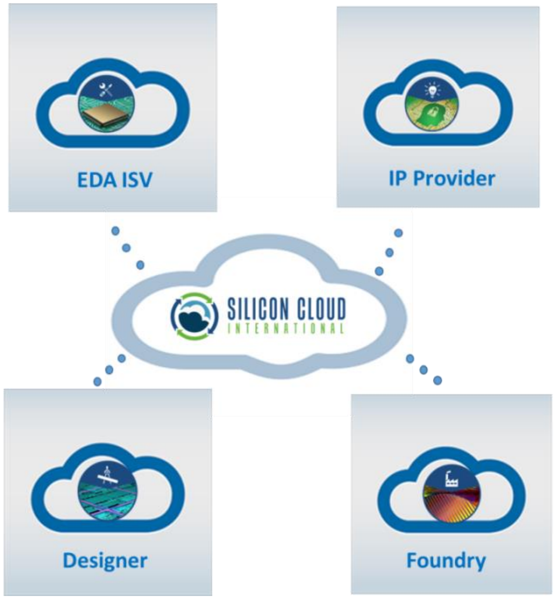
Thin SVC Client

U2F-Yubikey OTP
(2nd-Factor Authentication)

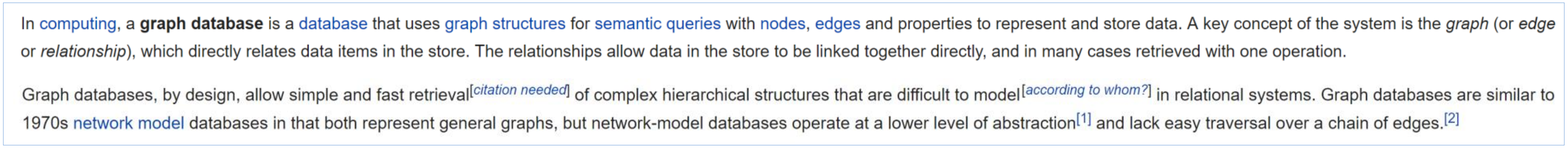
SCI Cloud Secure Inter-Organization Collaboration

- ***Workflow-engaged*** inter-organizational collaboration through Role-Based-Access-Control (RBAC)
  - *Digital Rights Management* (DRM) scheme is enforced through a ***"Vaulted Cloud"*** graph model wherein digital IP is *"execution provenanced"* in real time in a graph database.
    - Graph database of multi-propertied nodes and edges
  - Resulting graph is mined for pattern matches that workflow and DRM rules of behavior
    - Enforcement, security, monitoring, compliance
  - VMs are ***"execution provenanced"***
    - VMs do not exist as isolated computation engines, but have active database management during execution
  - Projects own VMs as *"computational objects"* that have life-cycled (cradle-2-grave) through a directed, acyclic graph-oriented workflow.
  - The ***Silicon Cloud*** encapsulates a ***WorkFlow-as-a-Service*** (WFaaS) and tightly-coupled ***Digital-Rights-Management-as-a-Service*** (DRMaaS) architecture.
- ***Ephemeral*** inter-organizational collaboration through SDN
  - Projects are data isolated at VLAN, TCP/IP, and VPN layers
  - Inter-organizational collaboration is supported through a provenance VM with data-isolated, computational object (VM)

Secure Multi-Organization Collaboration Cloud



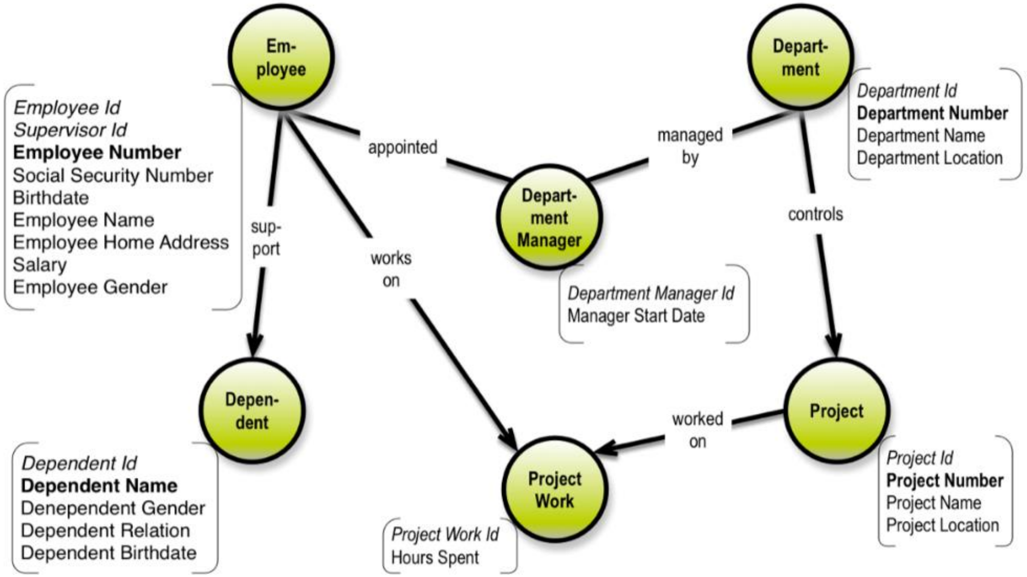
Graph Data Model For Workflow Compliance & IP Secure Collaboration (1)



In computing, a **graph database** is a database that uses graph structures for semantic queries with nodes, edges and properties to represent and store data. A key concept of the system is the *graph* (or *edge* or *relationship*), which directly relates data items in the store. The relationships allow data in the store to be linked together directly, and in many cases retrieved with one operation.

Graph databases, by design, allow simple and fast retrieval[*citation needed*] of complex hierarchical structures that are difficult to model[*according to whom?*] in relational systems. Graph databases are similar to 1970s network model databases in that both represent general graphs, but network-model databases operate at a lower level of abstraction[1] and lack easy traversal over a chain of edges.[2]

- ❑ Workflows resemble Directed Acyclic Graphs (**DAG**s), which are by name, **GRAPHS**, with associated **NODES** and **EDGES**
- ❑ Assigning *«property-attributes»* to NODES & EDGES results in a ***"property graph"***
- ❑ LCTC workflows are easily modeled as *"property graphs"*
  - ▪ NODES ➔ Workflow PHASE with *«node-properties»* as app used in that phase
  - ▪ EDGES ➔ IP transformation relationships between phases with «edge-properties» as dependency metadata



Graph Data Model For Workflow Compliance & IP Secure Collaboration (2)

- ❑ DEFINE the *workflow-reference-graph* (***WRG***) as the BASELINE for…
  - ❑ *Workflow Compliance Patterns* (***WCP***)

- ❑ *«execution provenance»* is transactionally recorded as a node-edge relationships in a graph database (***Titan*** and next ***DSE-Graph***)

- ❑ PROVENANCE TRAIL is created that TRACKS IP transformations across multi-phase workflow

- ❑ Repeated cycling of PHASES of the LCTC workflow can be defined as the "annealing" process of the workflow, i.e. the design is maturing…
  - ❑ (1) WITHIN a phase and (2) ACROSS ALL phases of the workflow

- ❑ Key element visibly apparent within the reference workflow DAG is the opportunity for IP PHASE EXTRACTION & INSERTION

- ❑ Encapsulating the entire workflow execution is a fundamental premise for cloud-based…
  - ▪ (1) workflow compliance, (2) IP collaboration, & (3) IP security

- ❑ Use Tinkerpop's ***Gremlin Query Language*** to search for relevant WCPs

Leveraging Virtualized & Software-Defined Infrastructure For Workflow Execution, Process & IP Provenance, & Security (1)

Defining Characteristics

❑ Engineering workflow applications are…

- (1) Performance-driven, (2) large memory footprint, (3) I/O intensive

❑ Infrastructure is embodied through bare-metal server farms

❑ Compute workloads are (1) dominantly batch-oriented & (2) interactive graphics (2D & 3D)

❑ Primarily SINGLE-threaded apps…MULTI-threaded apps are increasingly important

- Multi-threaded apps are more difficult to optimize performance across multi-core processors and INTER-server in generically networked server farms

❑ Linux desktops, e.g. Gnome, KDE, running an application's X-Windows GUI

❑ NFS file shares for inter-PHASE data sharing…**Object Store**?? 🡐 What is that?

❑ Data security is the FIREWALL appliance…INTERNAL data security (**RBAC**)? 🡐 What is that?

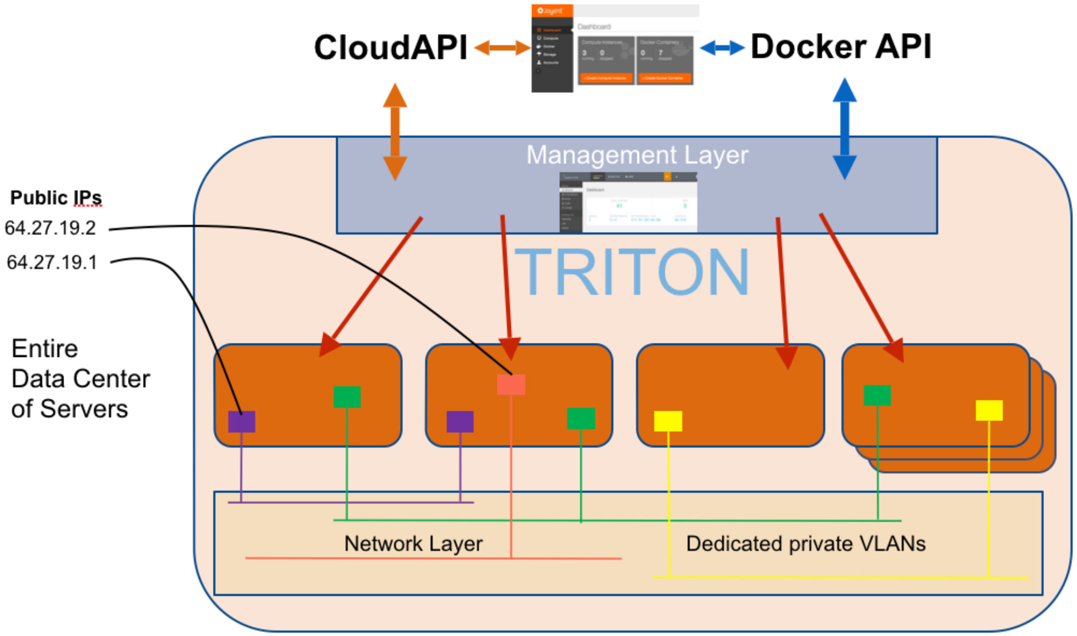
- **CHALLENGES FOR TRADITIONAL WORKFLOWS IN GENERALIZED PUBLIC CLOUDS**

❑ Transition of BATCH compute loads

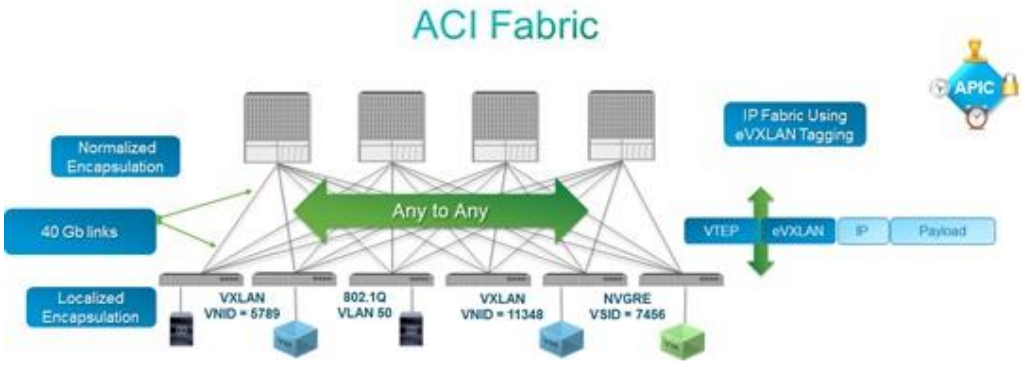
- FIXED bare-metal server farm with **LSF** 🡒 DYNAMICALLY provisioned & DYNAMICALLY-powered (vCPUs) Linux containers

❑ Move away from VNC desktop to JavaScript HTML-rendered <canvas> bitmap

Leveraging Triton Cloud Middleware & Cisco ACI Fabric Infrastructure For Security



- **Container virtualization and orchestration through open-source Triton/SmartOS Cloud Middleware**
- **Compute nodes ➔ Cisco C240 M4 (20C) Servers**
- **L2/L3/L4 Switching Fabric ➔**
- **Cisco ACI (Nexus 9K)**
    - **40G fabric backplane**



Leveraging Virtualized & Software-Defined Infrastructure For Workflow Execution, Process & IP Provenance, & Security (2)

TOWARDS HIGH PERFORMANCE VIRTUALIZATION

- ❑ Applications are virtualized through “application” & “multi-process” **Linux Containers** (**Docker API**)
- ❑ Containers as MICRO-serviced, MICRO-networked, MICRO-clustered ***“compute-shire”***
  - ▪ Provide a deterministic, flexible, socket-based micro-cluster for MULTI-threaded apps
  - ▪ 20x compute density improvement over fully virtualized OS VMs
- ❑ From bare metal server to **NON-hypervisor**, **OS-Virtualization** with **Linux Containers**
- ❑ FIXED bare-metal server farm with **LSF** → DYNAMICALLY provisioned & DYNAMICALLY-powered (vCPUs) Linux containers
  - ▪ 7 sec container provisioning
- ❑ Move from process-isolation to OS-user space isolation
- ❑ No virtio, containers have direct access to kernel-mode device drivers & adapters, e.g. **GPU**s
- ❑ Applications are virtualized through **Linux Containers** (**Docker API**)
- ❑ Containers as MICRO-serviced, MICRO-networked, MICRO-clustered ***compute-shire***
  - ▪ Provide a deterministic, flexible, socket-based micro-cluster for MULTI-threaded apps

Leveraging Virtualized & Software-Defined Infrastructure For Workflow Execution, Process & IP Provenance, & Security (3)

TOWARDS SOFTWARE-DEFINED-DATA-CENTER (SDDC)

- ❑ From fixed, DHCP-based networking ➔ *Software-Defined-Network* (**SDN)** (**Cisco ACI**)
  - VM-2-VM micro-networked, contract-oriented TCP flows
- ❑ Foundational *Software Defined Data Center* (**SDDC**) **VXLAN**, project-network isolation
- ❑ Integrate the container-orchestration with Cisco's leaf-spine ACI network
- ❑ ACI provides a RESTful, single point of control that is integrated with the container provisioning,
- ❑ *"Project SDDCs"* exist within a *"Tenant-of-Tenants"* (**ToT**) ACI architecture, i.e. *project-tenant*
- ❑ *"Project Access Containers"* (PACs) or are assigned as "computing identity" for EACH user
- ❑ *PACs* can only *"project connect"* to ONE *project-tenant* at a time and PACs are *"morphed"* with a project-owned PAC-filesystem that prevents removal of *project-owned IP* and *cross-project IP contamination*.
- ❑ Project IP sharing is facilitated through object-storage with ACI *"contracts"* between the object-store application and a project-network/VLAN.
- ❑ Node-Edge transactions are generated and recorded in graph database and Process or IP Compliance Violation Policies (CVP) are monitored against all IP Object Stores

Leveraging Virtualized & Software-Defined Infrastructure For Workflow Execution, Process & IP Provenance, & Security (4)

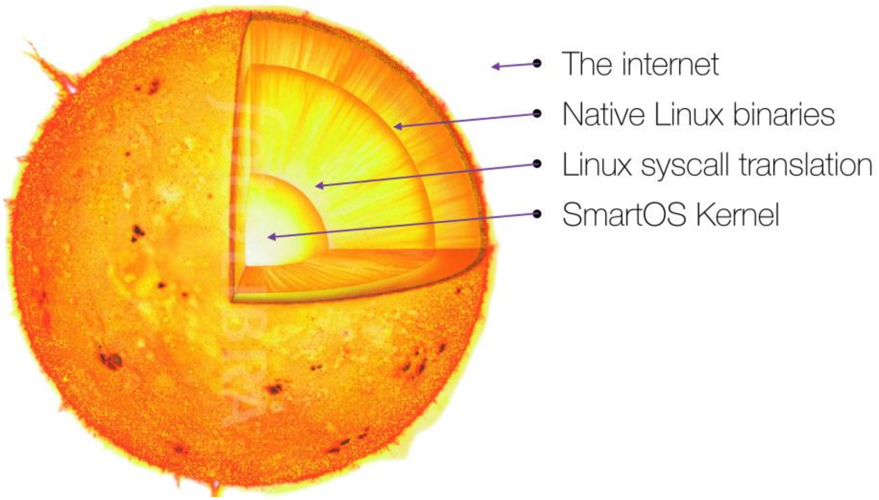
TOWARDS SOFTWARE-DEFINED-DATA-CENTER (SDDC)

❑ FIXED bare-metal server farm with LSF ➔ DYNAMICALLY provisioned & DYNAMICALLY-powered (vCPUs) Linux containers

❑ Applications are virtualized through Linux Containers (Docker API)

❑ Containers as MICRO-serviced, MICRO-networked, MICRO-clustered **compute-shire**

- Provide a deterministic, flexible, socket-based micro-cluster for MULTI-threaded apps



Leveraging Virtualized & Software-Defined Infrastructure For Workflow Execution, Process & IP Provenance, & Security (4)

OBJECT STORAGE WITH COMPUTE

❑ **CHALLENGES FOR TRADITIONAL WORKFLOWS IN GENERALIZED PUBLIC CLOUDS**

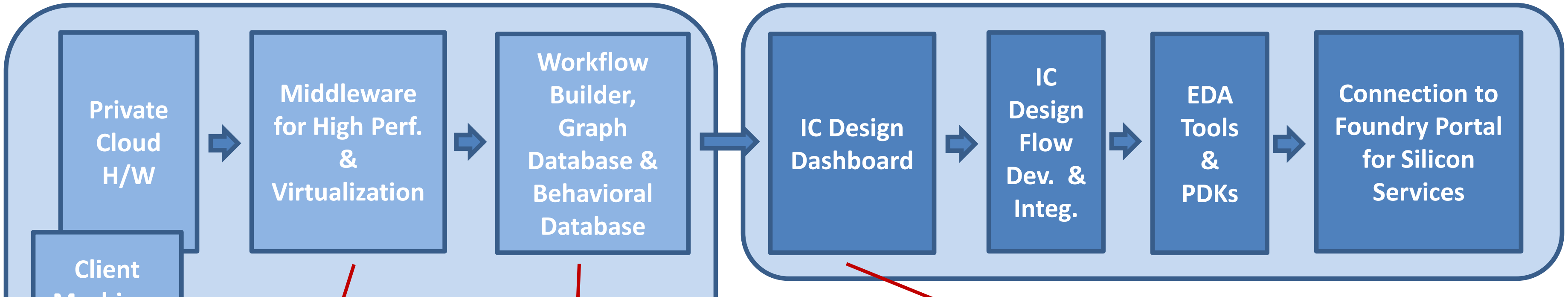
❑ Transition of BATCH compute loads

- FIXED bare-metal server farm with LSF ➔ DYNAMICALLY provisioned & DYNAMICALLY-powered (vCPUs) Linux containers

❑ Move from NAS to object-store-with-compute

❑ CLI phase/pipe processing for mapping & reducing *Data At Rest*

Summary of the SCI Cloud Technology Innovations



Private Cloud H/W

Client Machines System

Middleware for High Perf. & Virtualization

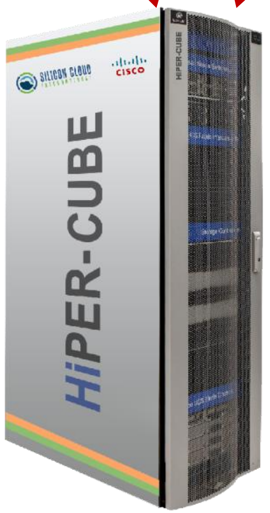
Workflow Builder, Graph Database & Behavioral Database

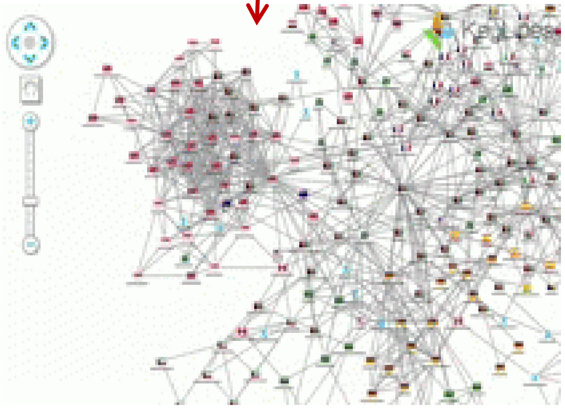
IC Design Dashboard

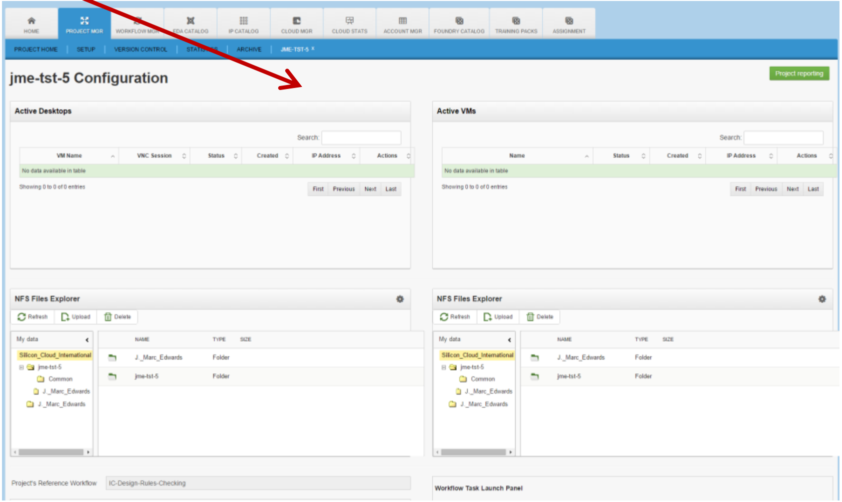
IC Design Flow Dev. & Integ.

EDA Tools & PDKs

Connection to Foundry Portal for Silicon Services







jme-tst-5 Configuration

Active Desktops

Search:

VM Name	VNC Session	Status	Created	IP Address	Actions
No data available in table					

Showing 0 to 0 of 0 entries

Active VMs

Search:

Name	Status	Created	IP Address	Actions
No data available in table				

Showing 0 to 0 of 0 entries

NFS Files Explorer

My data
- Silicon_Cloud_International
  - jme-tst-5
    - Common
    - J._Marc_Edwards
  - J._Marc_Edwards

NAME	TYPE	SIZE
J._Marc_Edwards	Folder	
jme-tst-5	Folder	

NFS Files Explorer

My data
- Silicon_Cloud_International
  - jme-tst-5
    - Common
    - J._Marc_Edwards
  - J._Marc_Edwards

NAME	TYPE	SIZE
J._Marc_Edwards	Folder	
jme-tst-5	Folder	

Project's Reference Workflow: IC-Design-Rules-Checking

Workflow Task Launch Panel

Key Technology Innovation

- Modern design workflows with user-execution-behavior database
- High performance compute hardware virtualization
- Advanced security & Workflow Management features

SILICON CLOUD INTERNATIONAL

NC STATE UNIVERSITY

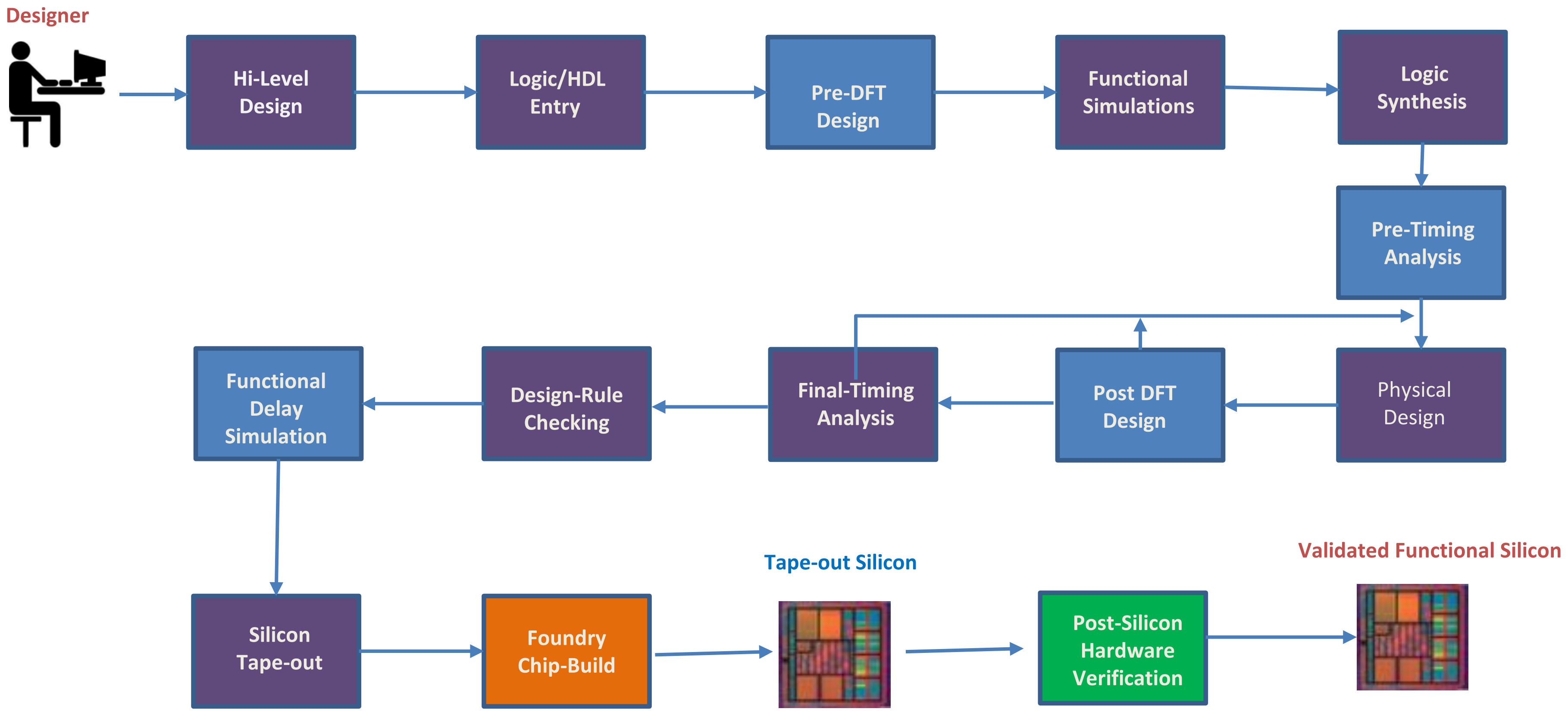
Workflow Compliance Example-1

NCSU Semiconductor design Workflow Application

Professor W. Rhett Davis (NCSU)

W. Rhett Davis consults for Silicon Cloud International (SCI) on the development of Semiconductor Workflows. He also conducts research in areas of interest similar to the business interest of SCI. The terms of this arrangement have been reviewed and approved by NC State University in accordance with its policy on objectivity in research.

Semiconductor Design Workflow Example



Designer

Hi-Level Design

Logic/HDL Entry

Pre-DFT Design

Functional Simulations

Logic Synthesis

Pre-Timing Analysis

Physical Design

Post DFT Design

Final-Timing Analysis

Design-Rule Checking

Functional Delay Simulation

Silicon Tape-out

Foundry Chip-Build

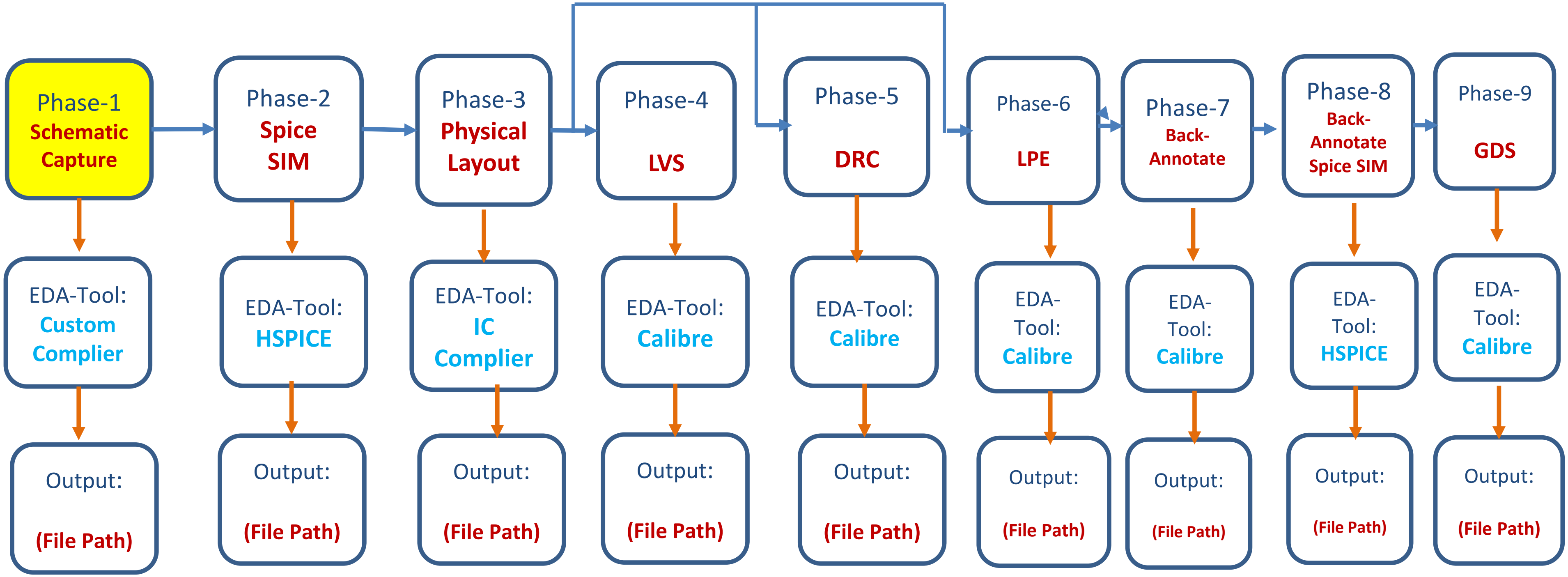
Tape-out Silicon

Post-Silicon Hardware Verification

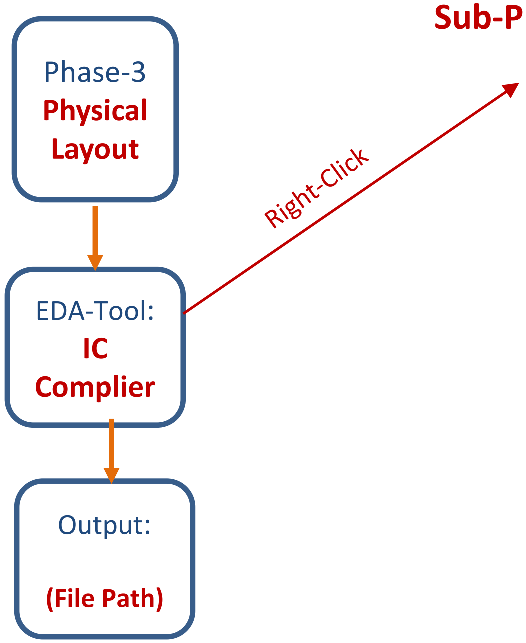
Validated Functional Silicon

Foundry Build

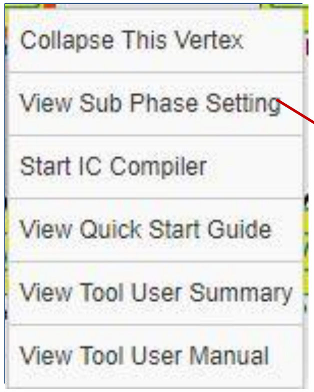
Overall X-Bar Design Workflow Graph



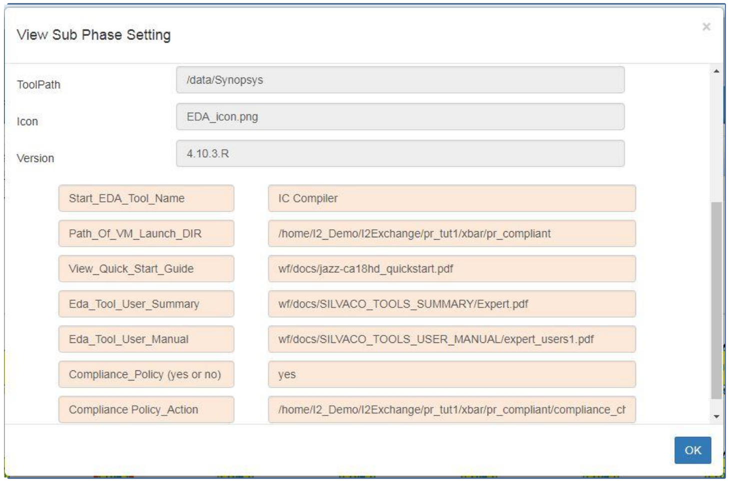
Sub-Phase Action & Property Setting Manuals



Sub-Phase Execution Action Selection Manual



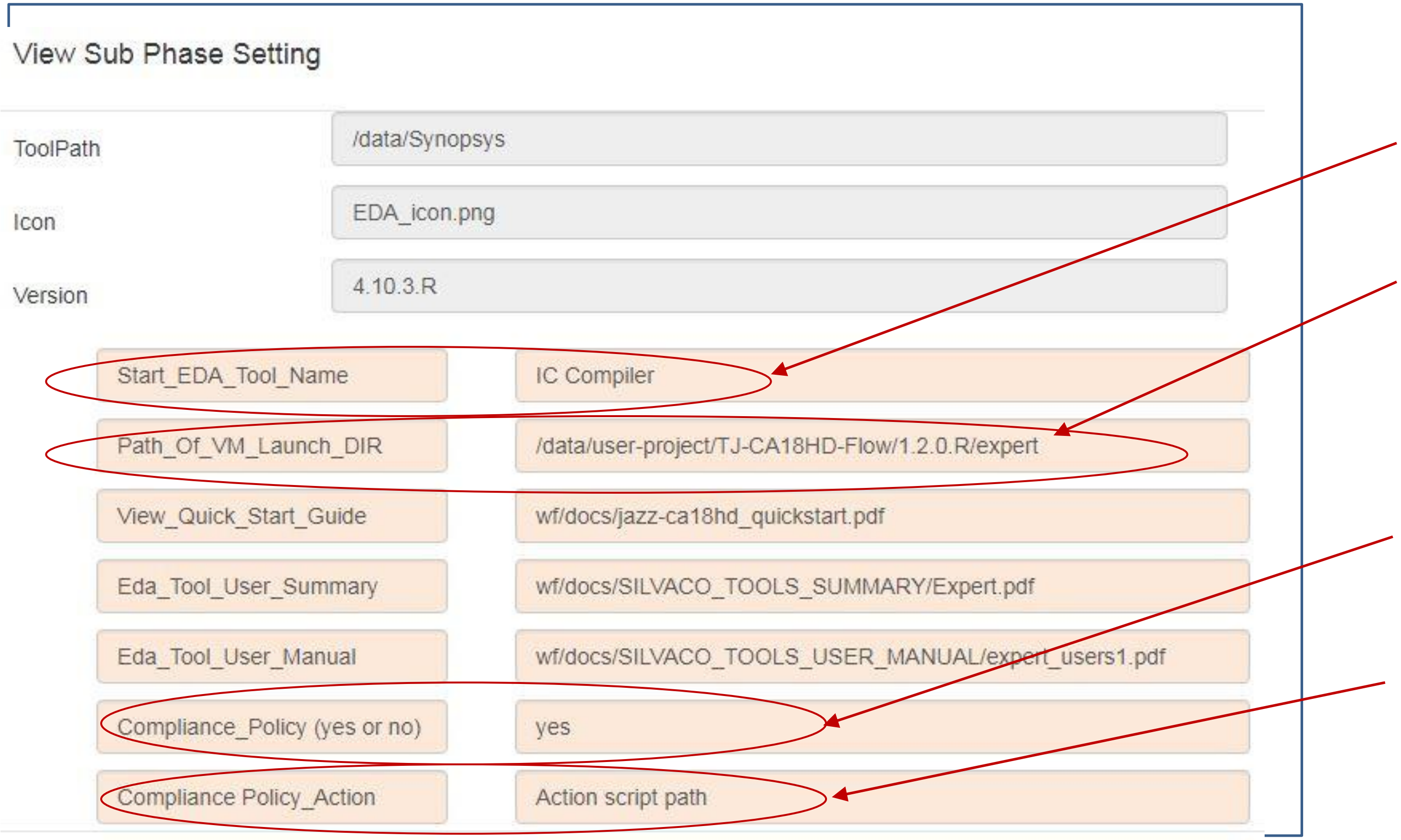
Sub-Phase Property Definition Manual



View Sub Phase Setting

ToolPath	/data/Synopsys
Icon	EDA_icon.png
Version	4.10.3.R
Start_EDA_Tool_Name	IC Compiler
Path_Of_VM_Launch_DIR	/home/I2_Demo/I2Exchange/pr_tut1/xbar/pr_compliant
View_Quick_Start_Guide	wf/docs/jazz-ca18hd_quickstart.pdf
Eda_Tool_User_Summary	wf/docs/SILVACO_TOOLS_SUMMARY/Expert.pdf
Eda_Tool_User_Manual	wf/docs/SILVACO_TOOLS_USER_MANUAL/expert_users1.pdf
Compliance_Policy (yes or no)	yes
Compliance Policy_Action	/home/I2_Demo/I2Exchange/pr_tut1/xbar/pr_compliant/compliance_ch

Sub-Phase Property Setting Manual (details)



View Sub Phase Setting

ToolPath	/data/Synopsys
Icon	EDA_icon.png
Version	4.10.3.R
Start_EDA_Tool_Name	IC Compiler
Path_Of_VM_Launch_DIR	/data/user-project/TJ-CA18HD-Flow/1.2.0.R/expert
View_Quick_Start_Guide	wf/docs/jazz-ca18hd_quickstart.pdf
Eda_Tool_User_Summary	wf/docs/SILVACO_TOOLS_SUMMARY/Expert.pdf
Eda_Tool_User_Manual	wf/docs/SILVACO_TOOLS_USER_MANUAL/expert_users1.pdf
Compliance_Policy (yes or no)	yes
Compliance Policy_Action	Action script path

EDA Tools Name

Working DIR Path

Compliance Policy (YES or NO select)

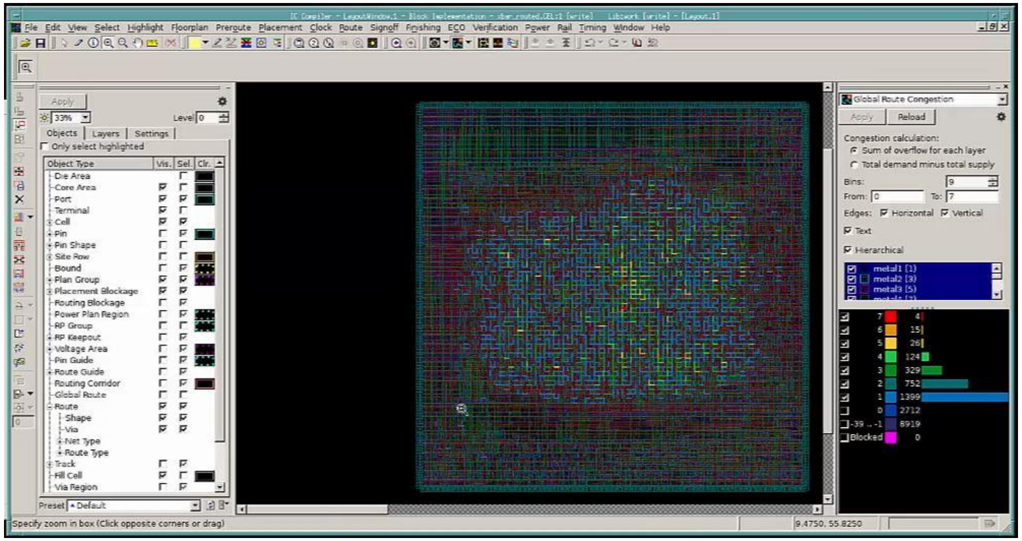
Compliance Policy Action (script path)

Semiconductor Design Workflow Compliance -- Demo-1

Design Process Compliance Check Example:

- ❑ To show a non-compliant design process case
  - Design cell layout placement and routing in conflict result
- ❑ To show a compliant design process case
  - Design cell layout placement and routing in non-conflict result

Non-Compliant Layout Case


File Edit View Select Highlight Floorplan Preroute Placement Clock Route Signoff Finishing ECO Verification Power Rail Timing Window Help
Apply
Objects Layers Settings
Object Type Vis. Sel. Clr.
Preset Default

Global Route Congestion

Apply Reload

Congestion calculation:
● Sum of overflow for each layer
○ Total demand minus total supply

Bins: 9

From: 0 To: 7

Edges: ☑ Horizontal ☑ Vertical

☑ Text

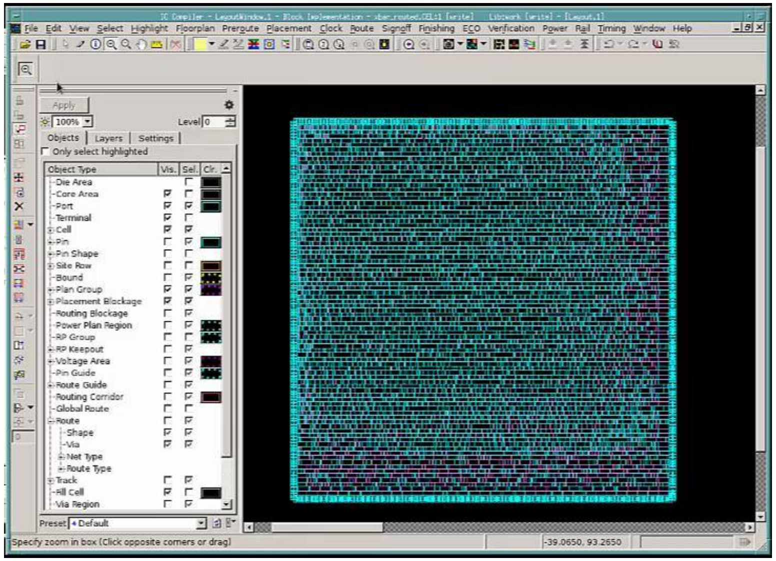
☑ Hierarchical

☑ metal1 (1)
☑ metal2 (3)
☑ metal3 (5)

☑	7	🟥	4
☑	6	🟧	15
☑	5	🟨	26
☑	4	🟩	124
☑	3	🟩	329
☑	2	🟦	752
☑	1	🟦	1399
☐	0	🟦	2712
☐	-39 .. -1	🟪	8919
☐	Blocked	🟪	0

Specify zoom in box (Click opposite corners or drag) 9.4750, 55.8250

Compliant Layout Case


File Edit View Select Highlight Floorplan Preroute Placement Clock Route Signoff Finishing ECO Verification Power Rail Timing Window Help
Apply
Objects Layers Settings
Object Type Vis. Sel. Clr.
Preset Default

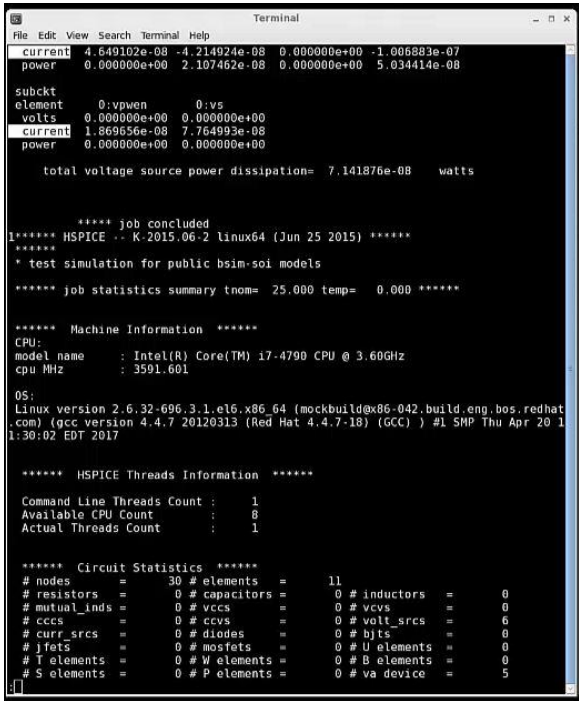
Specify zoom in box (Click opposite corners or drag) -39.0650, 93.2650

Semiconductor Design Workflow Compliance -- Demo-2

HSPICE Simulation Model Standard Compliance Check Example:

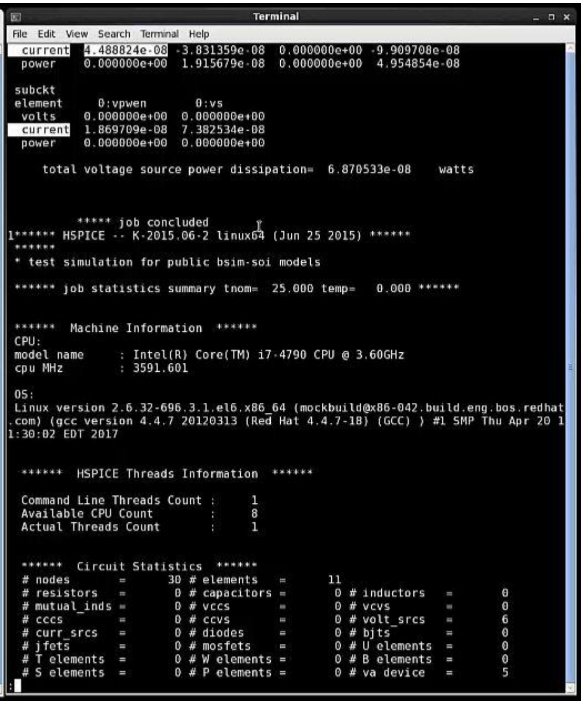
- □ To show a non-compliant simulation model case
  - Result device leakage current 2% above standard specification
- □ To show a compliant simulation model case
  - Result device leakage current within the standard specification

Non-Compliant Result Case



```
  current   4.649102e-08 -4.214924e-08  0.000000e+00 -1.006883e-07
  power     0.000000e+00  2.107462e-08  0.000000e+00  5.034414e-08

 subckt
 element      0:vpwen        0:vs
  volts     0.000000e+00  0.000000e+00
  current   1.869656e-08  7.764993e-08
  power     0.000000e+00  0.000000e+00

     total voltage source power dissipation=  7.141876e-08     watts



          ***** job concluded
1****** HSPICE -- K-2015.06-2 linux64 (Jun 25 2015) ******
 ******
 * test simulation for public bsim-soi models

 ****** job statistics summary tnom=  25.000 temp=   0.000 ******


 ******  Machine Information  ******
 CPU:
 model name      : Intel(R) Core(TM) i7-4790 CPU @ 3.60GHz
 cpu MHz         : 3591.601

 OS:
 Linux version 2.6.32-696.3.1.el6.x86_64 (mockbuild@x86-042.build.eng.bos.redhat
.com) (gcc version 4.4.7 20120313 (Red Hat 4.4.7-18) (GCC) ) #1 SMP Thu Apr 20 1
1:30:02 EDT 2017


  ******  HSPICE Threads Information  ******

  Command Line Threads Count :     1
  Available CPU Count        :     8
  Actual Threads Count       :     1


  ******  Circuit Statistics  ******
  # nodes       =      30 # elements   =      11
  # resistors   =       0 # capacitors =       0 # inductors   =       0
  # mutual_inds =       0 # vccs       =       0 # vcvs        =       0
  # cccs        =       0 # ccvs       =       0 # volt_srcs   =       6
  # curr_srcs   =       0 # diodes     =       0 # bjts        =       0
  # jfets       =       0 # mosfets    =       0 # U elements  =       0
  # T elements  =       0 # W elements =       0 # B elements  =       0
  # S elements  =       0 # P elements =       0 # va device   =       5
:
```

Compliant Result Case



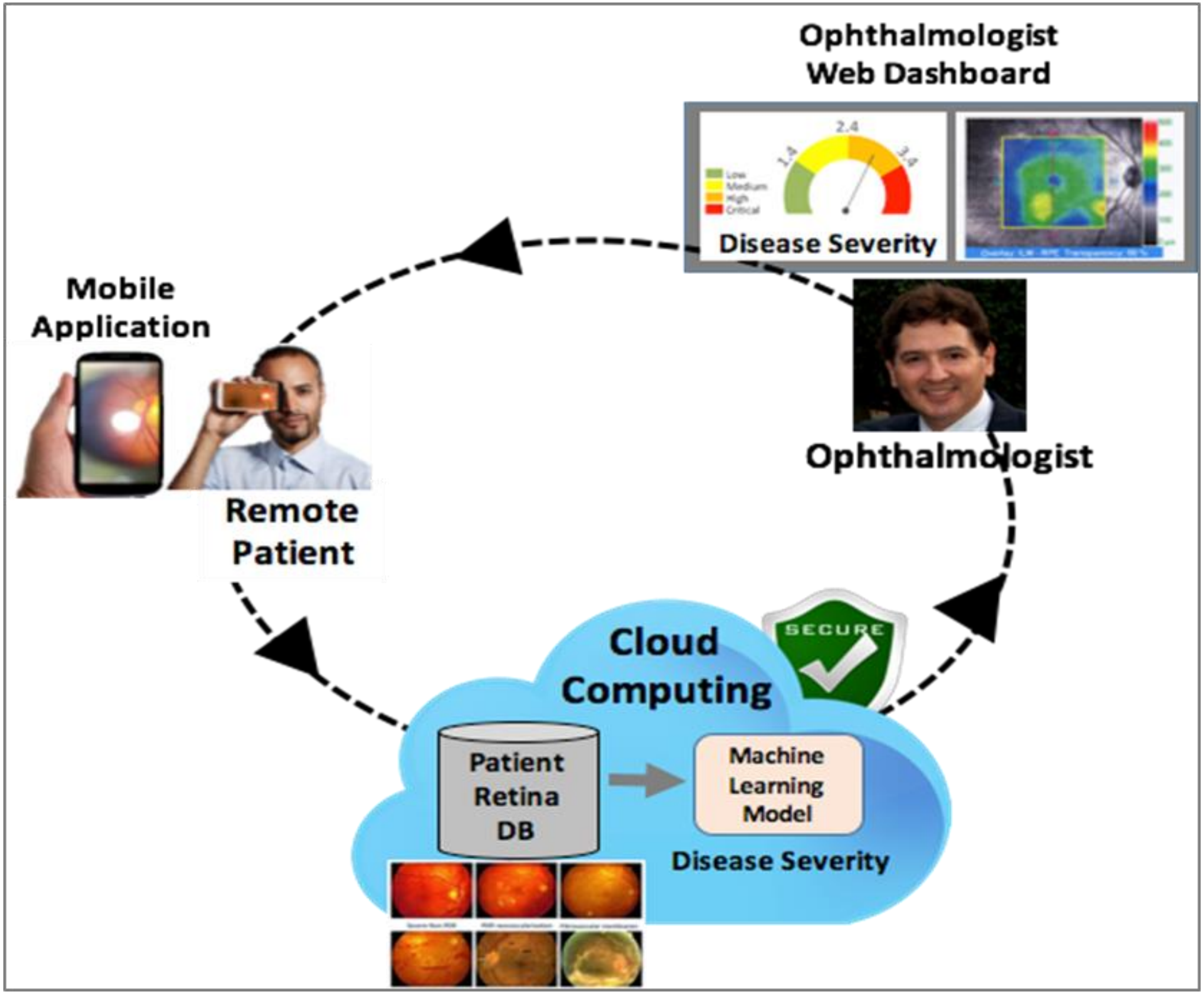
```
  current   4.488824e-08 -3.831359e-08  0.000000e+00 -9.909708e-08
  power     0.000000e+00  1.915679e-08  0.000000e+00  4.954854e-08

 subckt
 element      0:vpwen        0:vs
  volts     0.000000e+00  0.000000e+00
  current   1.869709e-08  7.382534e-08
  power     0.000000e+00  0.000000e+00

     total voltage source power dissipation=  6.870533e-08     watts



          ***** job concluded
1****** HSPICE -- K-2015.06-2 linux64 (Jun 25 2015) ******
 ******
 * test simulation for public bsim-soi models

 ****** job statistics summary tnom=  25.000 temp=   0.000 ******


 ******  Machine Information  ******
 CPU:
 model name      : Intel(R) Core(TM) i7-4790 CPU @ 3.60GHz
 cpu MHz         : 3591.601

 OS:
 Linux version 2.6.32-696.3.1.el6.x86_64 (mockbuild@x86-042.build.eng.bos.redhat
.com) (gcc version 4.4.7 20120313 (Red Hat 4.4.7-18) (GCC) ) #1 SMP Thu Apr 20 1
1:30:02 EDT 2017


  ******  HSPICE Threads Information  ******

  Command Line Threads Count :     1
  Available CPU Count        :     8
  Actual Threads Count       :     1


  ******  Circuit Statistics  ******
  # nodes       =      30 # elements   =      11
  # resistors   =       0 # capacitors =       0 # inductors   =       0
  # mutual_inds =       0 # vccs       =       0 # vcvs        =       0
  # cccs        =       0 # ccvs       =       0 # volt_srcs   =       6
  # curr_srcs   =       0 # diodes     =       0 # bjts        =       0
  # jfets       =       0 # mosfets    =       0 # U elements  =       0
  # T elements  =       0 # W elements =       0 # B elements  =       0
  # S elements  =       0 # P elements =       0 # va device   =       5
:
```

Workflow Compliance Example-2

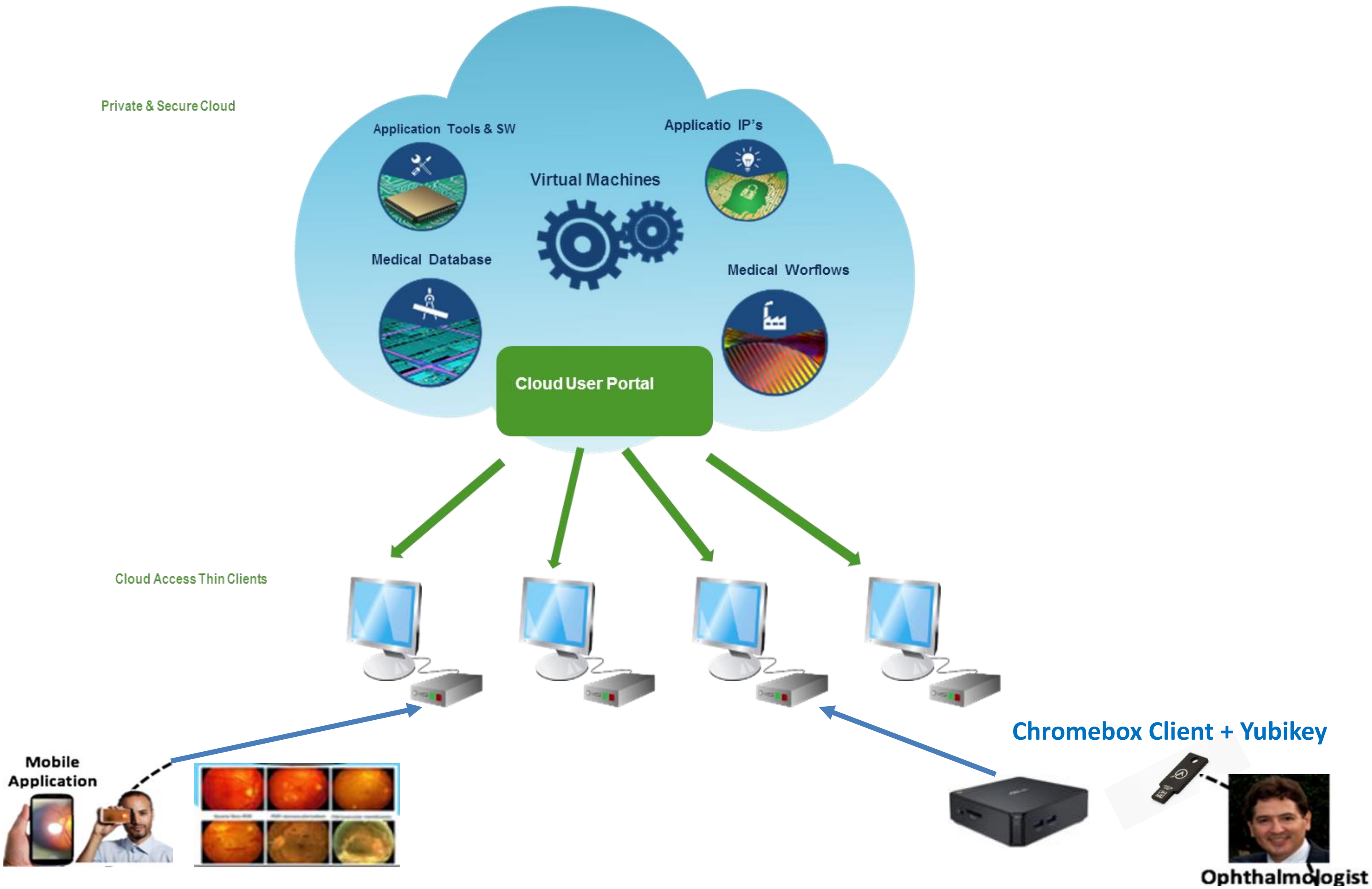
UTSA Cloud-Based Secure Mobile Teleophthalmology Healthcare Application

Professor Paul Rad (UTSA)

Cloud-Based Secure Mobile Teleophthalmology Workflow



Silicon Cloud Integration of the Teleophthalmology Workflow



Private & Secure Cloud

Application Tools & SW

Applicatio IP's

Virtual Machines

Medical Database

Medical Worflows

Cloud User Portal

Cloud Access Thin Clients

Mobile Application

Chromebox Client + Yubikey

Ophthalmologist

Mobile App tokes & encrypts fundus Images & uploads to Cloud

Doctor Login to Medical Dashboard in Cloud

Teleophthalmology Workflow Processes

Step 1: Taking Picture from Patient’s Fundus in different locations

Step 2: The mobile-cloud App will encrypt the Taken Fundus Image and will send the encrypted image to the secure multitenant cloud storage

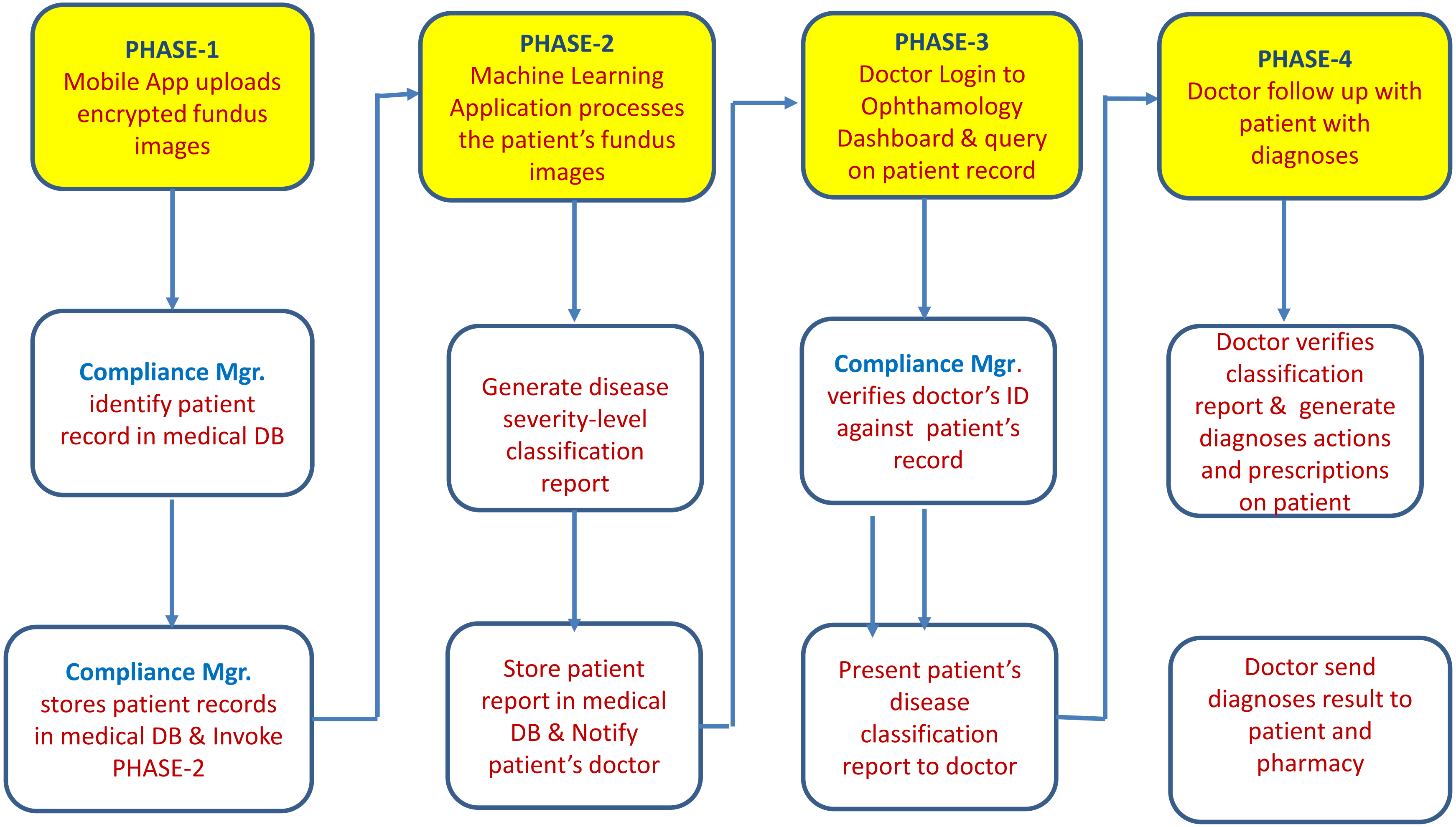
Step 3: The Machine Learning Algorithm that is designed to identify patient disease severity level runs on all collected fundus images stored in the secure multitenant cloud storage.

Step 4: System will classifies patients based on their disease severity level and results will be send to the Ophthalmologist dashboard.

Step 5: When Ophthalmologist login to his cloud dashboard, list of patients and their disease severities as well as their fundus image is presented.

Step 6: Ophthalmologist can follow up with patients on follow ups and diagnoses

Teleophthalmology Workflow Execution Graph (with AI-Compliance Management)



PHASE-1
Mobile App uploads encrypted fundus images

Compliance Mgr.
identify patient record in medical DB

Compliance Mgr.
stores patient records in medical DB & Invoke PHASE-2

PHASE-2
Machine Learning Application processes the patient's fundus images

Generate disease severity-level classification report

Store patient report in medical DB & Notify patient's doctor

PHASE-3
Doctor Login to Ophthamology Dashboard & query on patient record

Compliance Mgr.
verifies doctor's ID against patient's record

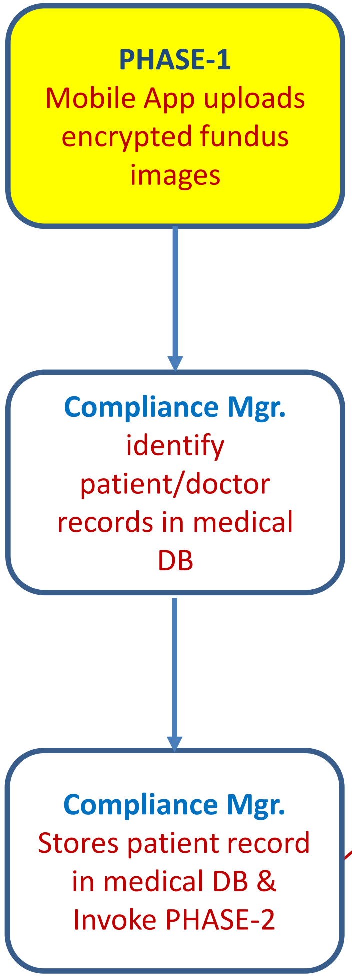
Present patient's disease classification report to doctor

PHASE-4
Doctor follow up with patient with diagnoses

Doctor verifies classification report & generate diagnoses actions and prescriptions on patient

Doctor send diagnoses result to patient and pharmacy

Phase-1 & Sub-Phase Property Definition Example

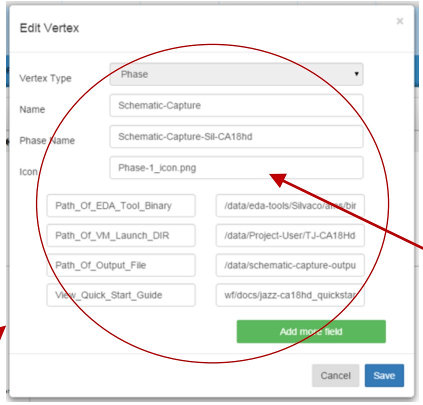


PHASE-1
Mobile App uploads encrypted fundus images

Compliance Mgr.
identify patient/doctor records in medical DB

Compliance Mgr.
Stores patient record in medical DB & Invoke PHASE-2

Phase-Property Definition Manual



Edit Vertex

Vertex Type: Phase

Name: Schematic-Capture

Phase Name: Schematic-Capture-Sil-CA18hd

Icon: Phase-1_icon.png

Path_Of_EDA_Tool_Binary	/data/eda-tools/Silvaco/ams/bir
Path_Of_VM_Launch_DIR	/data/Project-User/TJ-CA18Hd
Path_Of_Output_File	/data/schematic-capture-outpu
View_Quick_Start_Guide	wf/docs/jazz-ca18hd_quickstar

Phase-Property Action Definitions:

- **Identify patient's name & ID**
- **Search Patient's file in DB**
- **Identify patient's doctor**
- **Compliance Manager verify patient /doctor association records**
- **Compliance Manager stores patient 's images in his DB record.**
- **Invoke next Phase action path**