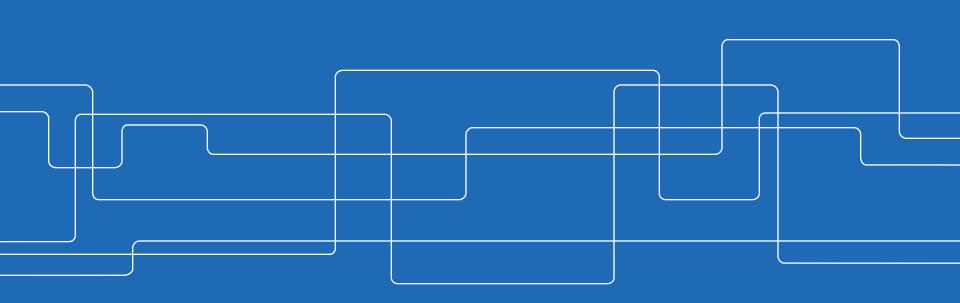


## Search-Based Design of Large Software Systems-of-Systems

Robert Lagerström, Pontus Johnson, and Mathias Ekstedt KTH Royal Institute of Technology





### Outline

Today:

Introduction to a systems-of-systems modeling and analysis tool

Search-based software engineering

Tomorrow:

Search-based design of systems-of-system?

## Our EA tool



### **Enterprise Architecture Analysis Tool**



Industrial Information and Control Systems Object Modeler

### A demo... (Download at: www.ics.kth.se/eaat)

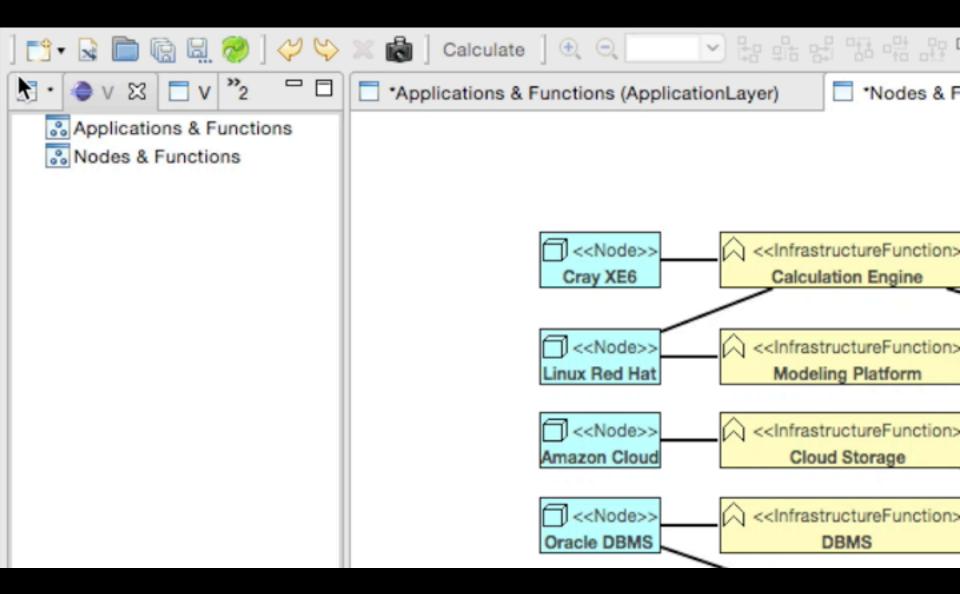
# Adding Applications

é c	QuickTime Player	Arkiv	Redigera	Innehåll	Fönste	r Hjälp				
8 - 8								E	AAT Objec	t M
] 🔁 - 🖻	🛅 🗟 🗟 🤣 ] 🖉	े × 🕯	Calculate	] 🔍 🔍	100% ~		<u>.</u> ?? 🔁	면	Snap To Grid	d
<b>O</b>	🔷 Vi 🗋 Na 📋	Vi	x 🗆 🗖	View 1	23					
▼untitled	.iEaat (MAP2 beta veaat)									
▼Clas	sses									
1	AccuracyConstraint									
	ApplicationCollaboration									
	ApplicationComponent									
	ApplicationFunction									
	ApplicationInterface									
	ApplicationRequirement									
	ApplicationService Artifact									
	AvailabilityConstraint BusinessActor									
	BusinessCollaboration									
	BusinessEvent									
	BusinessInterface									
1	BusinessObject									
	BusinessProcess									
1	BusinessRole									
1	BusinessService									
(	CapexConstraint									

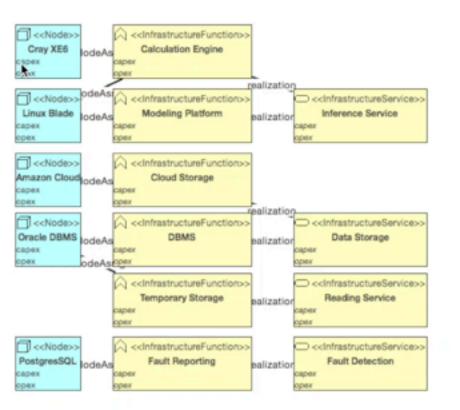
## Adding Infrastructure

Property V Filters Attributes Link Label Link Role N Movesble I ObjectConn V Nisc Mode	Value false false false
ح ۵۰ ۲	3. ~ ~ 0
	- C

### Cost View



# Adding Costs

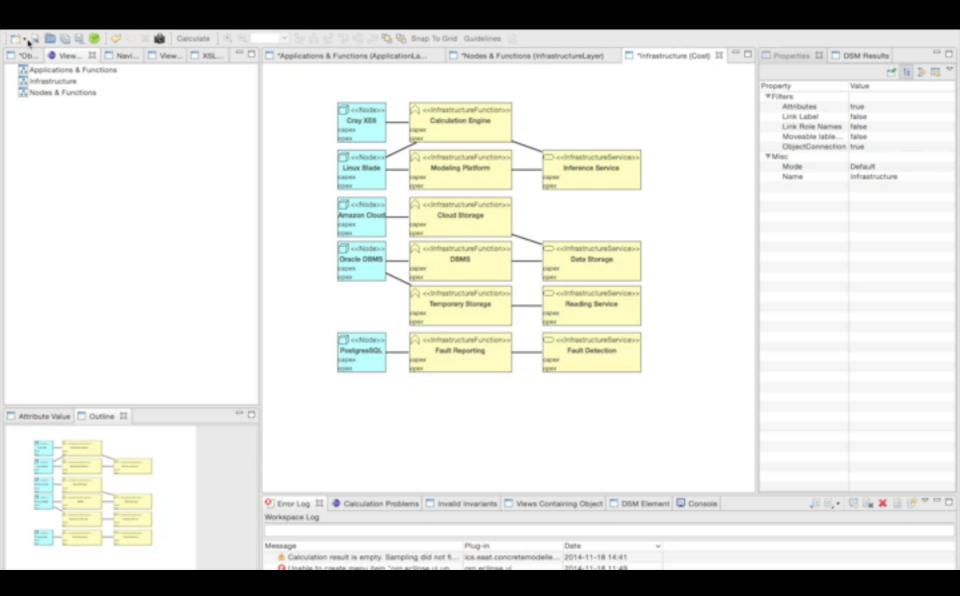


		- E	ŧ
Property	Value		
▼Filters			
Attributes	true		
Link Label	true		
Link Role Names	false		
Moveable lable	false		
ObjectConnection	true		
Misc			

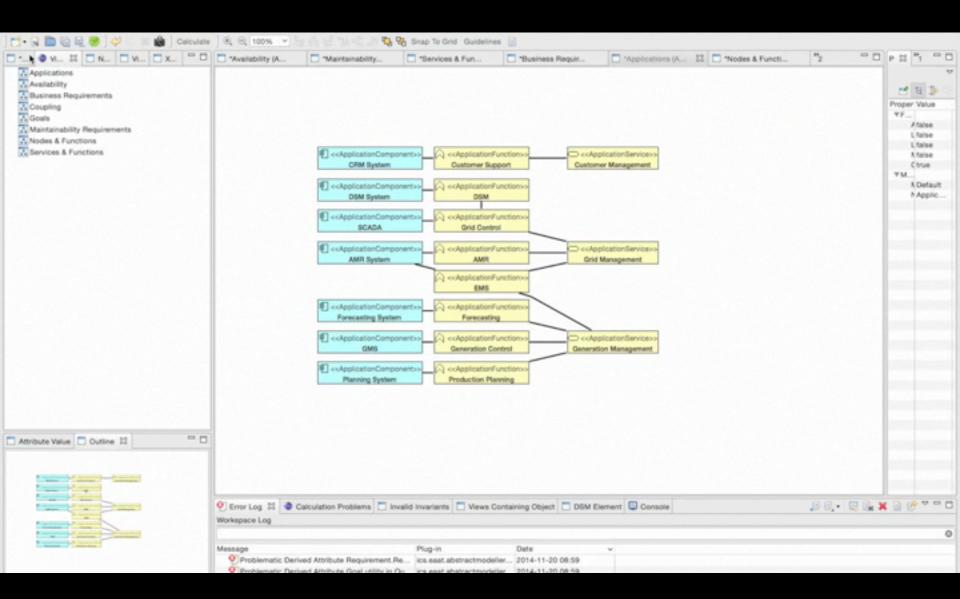
### Calculating Infrastructure Costs

CuickTime Player	Arkiv	Redigera	Innehåll	Fönster	Hjälp		
						EAAT (	Dbject Modele
] 📬 🗟 🛅 🗟 🗟 🔗 ] 💝	$\odot \times \mathbf{i}$	Calculate	1	00% 🖌 🖁		양금 ~음 _음 다 다	Snap To Grid
- Ob Ob 🕲 🗖 Na	avi	View 🗖 XS	SL 🗖 🗖	-Applic	ations &	Functions (Applicati	onLayer)
Applications & Functions Nodes & Functions View 3				•		Cray XE6 capex opex Cray XE6 capex opex Capex	Calculation Calculation Calculation Capex opex Opex Capex opex Cloud S capex opex Cloud S capex opex Cloud S capex opex

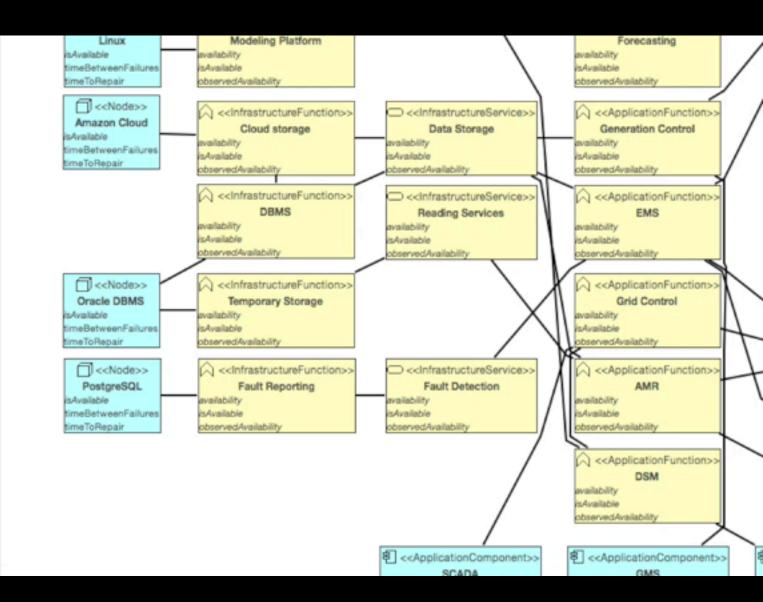
### Calculating Application Costs



## Calculating Availability

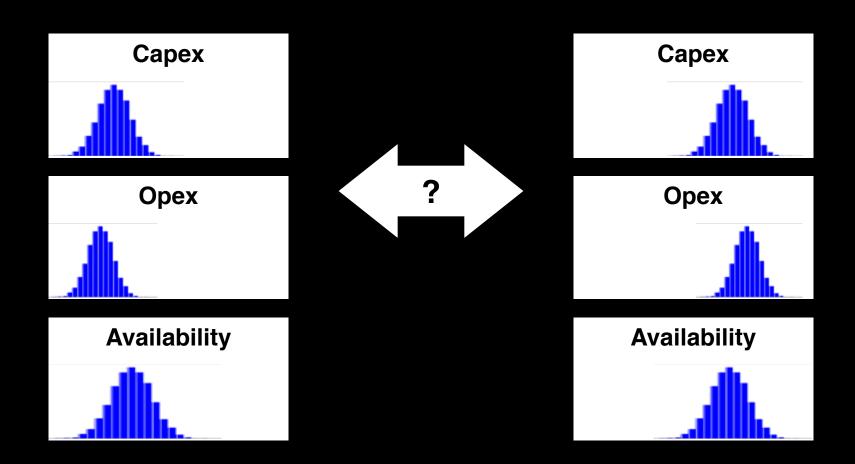


## Introducing Redundancy



- -

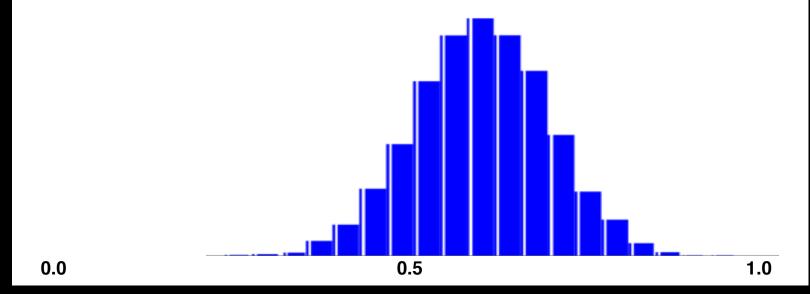
# Design Trade-offs



# Specifying Requirements

	alculate ] 🔍 🔍			Sna
	Coupling (C	oupling)	Goals (Goals)	🗖 •Mai
<ul> <li>Applications</li> <li>Availability</li> <li>Business Requirements</li> <li>Coupling</li> <li>Goals</li> <li>Maintainability Requirements</li> <li>Nodes &amp; Functions</li> <li>Services &amp; Functions</li> </ul>	<i>isAvai</i> timeB	Node>> Cray XE6 lable SetweenFailures ToRepair		Engine

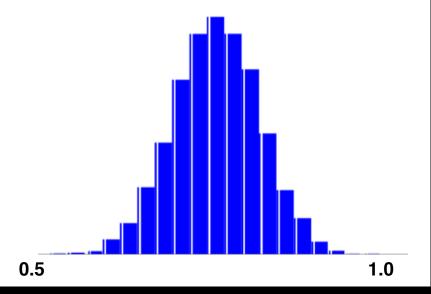
#### CustomerManagementReqs.utility



## Prioritizing Requirements

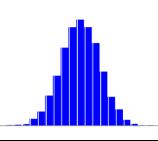
] 📬 🛃 🛅 🕼 🖳 🤣 ] 💛 🛇 🐹 📸 ] Calculate ] 🔍 🔍 100% 🕑 🚼 🖧 🖽 🖽 🖽	<mark>ሚ ዊ</mark> Sna
Vi Vi Vi X Vi X	
<pre></pre>	int 2

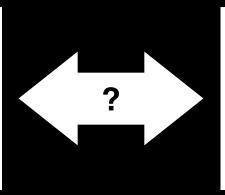
### **OperationalQuality.utility**



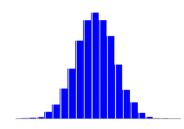
### Design Trade-off Aggregation

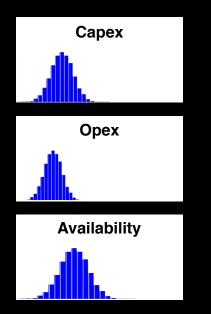
#### **OperationalQuality.utility**

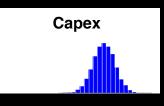


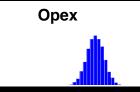


#### OperationalQuality.utility













#### **Current quality attribute coverage**

Requirements (utility) Application service Availability Application service cost Data accuracy Application component coupling Application size

System interoperability

Information security



#### Search-Based Software Engineering

**Optimization techniques:** 

Hill climbing Genetic algorithms Simulated Annealing Particle swarm optimization Ant colony optimization

> Mark Harman, Bryan F. Jones, Search-based software engineering, Information and Software Technology, 2001



Information and Software Technology 43 (2001) 873-879

AND SOFTWAR TECHNOLOGY

INFORMATION

#### Search-based software engineering

#### Mark Harman1.8, Bryan F. Jones11

"Department of Information Systems and Computing, Brand University, Editidge, Middlews, EBH 3PH, EK School of Computing, Detrovity of Genergon, Presprint, CF37 104, UK

#### Abstract

This paper claims that a new little of software engineering research and gravitor is enorging search-based software engineering. The paper argues that software origineering is ideal for the application of membraristic search techniques, such as genetic algorithms, simuland atoraling and also search. Such search-based inclusions could provide solutions to the difficult problems of balancing competing (and some times inconsistent) constraints and may suggest ways of finding acceptable solutions in situations where perfect solutions are either heurically impossible or practically infrashie

In order to develop the field of watch-based software engineering, a softwarelation of classic software engineering problems as watch mblems is required. The paper briefly sets out key ingendients for suscenabla reformulation and evaluation estimate sensets based software engineering, @ 2001 Elsevier Science R.V. All rights mean-od.

Rywork Silven opinoring. Middeutsis; Onetic algorithm

#### 1. Introduction

Software engineers often face problems associated with the balancing of competing constraints, trade-offs between concerns and requirement improvision. Perfact solutions are often either impossible or impractical and the nature of the undelesses often makes the definition of analytical algorithms. and longing

Like other registering disciplines, software registerin is typically concerned with near optimal solutions or those which fall within a specified acceptable inferance. It is precisely three factors which make robust metabraristic search based optimisation techniques readily applicable.

Metabouristic algorithms, such as genetic algorithms. (GA) [17], simulated assocaling [37] and take search [14] have been applied successfully to a number of engineering problems. For example, a literature survey of genetic algorithms reveals applications to, among others:

- mechanical engineering [2437]
- chemical engineering (6.22)
- medical and biomedical engineering [29,30,40]
- stvil engineering [1A:14,19]
- electronic engineering [5,11,25]

\* Companying asket Tel.: + 01/898.276.000 fax: + 02/898.281

E-mail addressor matchematiltranelarsk (M. Harman), Maanifgies and (R.F. Jawa) Tel: +01107-00710 Bev +011023-00718

1855-544401-6 - un fran mater @ 2011 Elsevier Science B-V. All rights merced PD 50910.5849(01)00189-6

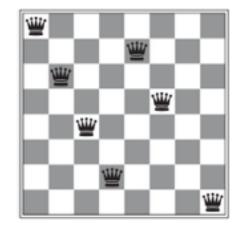
GA research and researchers have even monived into from observers in the field of social science. Through GA practitioners may not sense with the fastings of sociologists [18], it is an indication of the wide appreciation of the significance of these search-based technologies that they should have penetrated the collective consciousness of even 'non-technical' disciplines such as social science.

However, the discipline of software engineering appears to be unique with regard to the application of genetic algorithms (and similar search-based, metaheuristic optimisa tion techniques); metaheuristic algorithms have received comparatively little attention from software engineers in comparison with that which they have motived from researchers and practitioners in the more established fields of engineering.

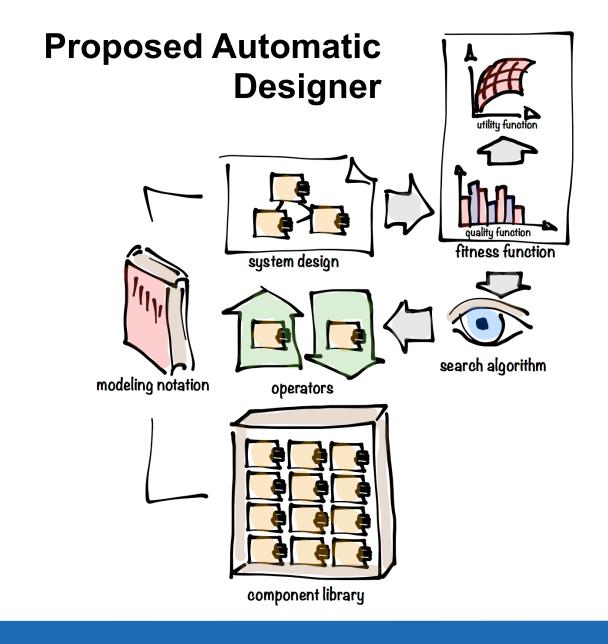
A literature search on 'software engineering' and 'genetic algorithms' groups work within the areas of testing [20.21.36.28.38.39] and cost estimation [12.13], but few others. Work has also taken place on automatic program ming using genetic programming (23) and parallelisation [31,44], which could be thought of specific topics within ording, which in turn, could be considered to be a part of software engineering. Even with the inclusion of this work on coding, the application of search-based inclusiones to problems in arithware engineering has been, hitlerics, somewhat patchy. The thesis underpinning the present paper is that search based metabeuristic optimisation inclusions are highly applicable to software engineering and that their

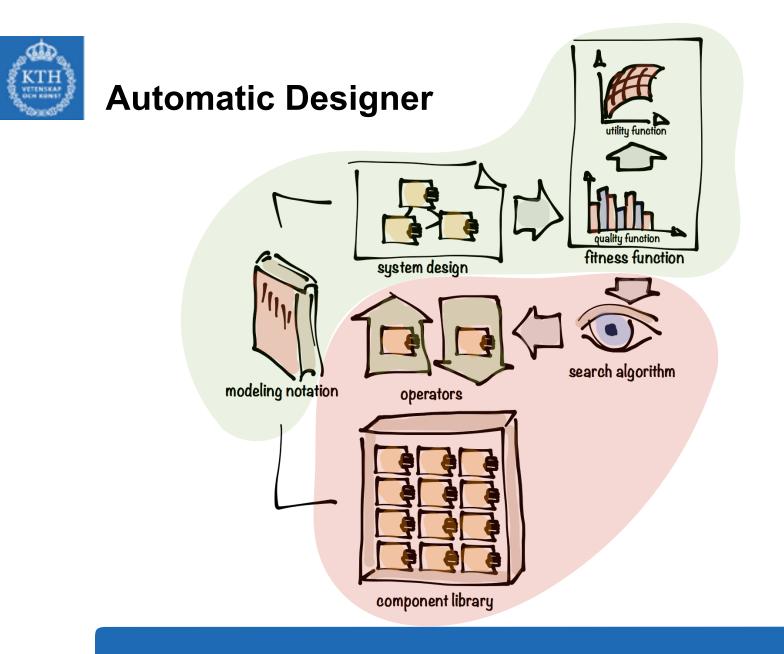


### Definition of a (search) problem



- The initial state that the agent starts in.
- A description of the possible actions available to the agent.
- A description of what each action does; the transition model.
- The **fitness function/goal test** is the characterization of what is considered to be a good solution.
- A path cost function that assigns a numeric cost to each path.







### Joint Utility Function

