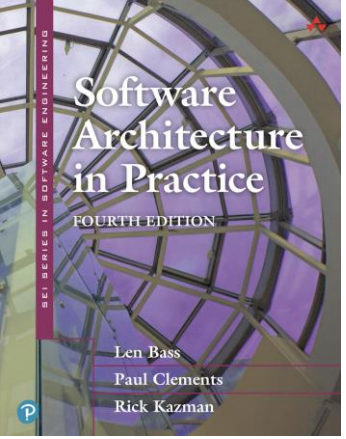
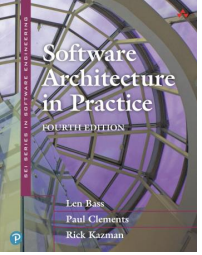


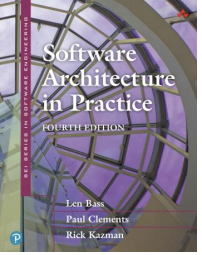
Chapter 6: Energy Efficiency

Energy is a bit like money: If you have a positive balance, you can distribute it in various ways, but according to the classical laws that were believed at the beginning of the century, you weren't allowed to be overdrawn.
—Stephen Hawking



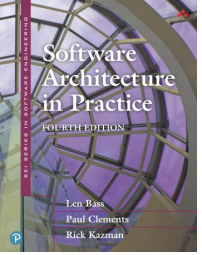
Chapter Outline

- What is Energy Efficiency?
- Energy Efficiency General Scenario
- Tactics for Energy Efficiency
- Tactics-Based Questionnaire for Energy Efficiency
- Patterns for Energy Efficiency
- Summary



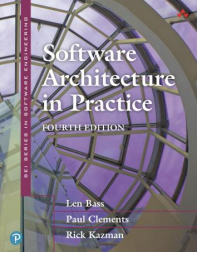
What is Energy Efficiency?

- Energy used by computers used to be free and unlimited—or at least that's how we behaved. Architects rarely worried about energy consumption of software in the past.
- But with the dominance of mobile devices, the increasing adoption of the IoT, and the ubiquity of cloud services, energy has become an issue that architects can no longer ignore.
- The consumption of energy can be managed by the architect, as with any other QA.



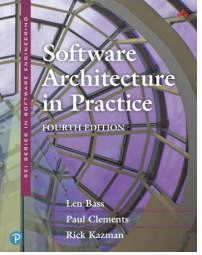
Energy Efficiency Concerns

- An architectural approach is necessary to gain control over *any* important system quality attribute, and energy efficiency is no different. If system-wide techniques for monitoring and managing energy are lacking, then developers are left to invent them on their own.
- Most architects and developers are unaware of energy efficiency as a QA of concern, and hence do not know how to go about engineering and coding for it.
- Most architects and developers lack suitable design concepts—models, patterns, tactics, and so forth—for designing for energy efficiency, as well as managing and monitoring it at runtime.



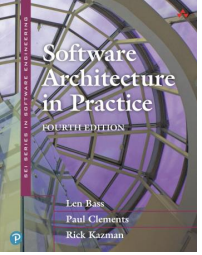
Energy Efficiency General Scenario

Portion of Scenario	Description	Possible Values
Source	This specifies who or what requests or initiates a request to conserve or manage energy.	End user, manager, system administrator, automated agent
Stimulus	A request to conserve energy.	Total usage, maximum instantaneous usage, average usage, etc.
Artifacts	This specifies what is to be managed.	Specific devices, servers, VMs, clusters, etc.
Environment	Energy is typically managed at runtime, but many interesting special cases exist, based on system characteristics.	Runtime, connected, battery-powered, low-battery mode, power-conservation mode



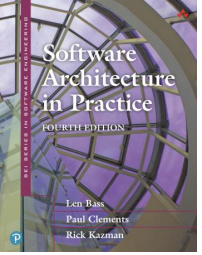
Energy Efficiency General Scenario

Response	What actions the system takes to conserve or manage energy usage.	One or more of the following: ▪ Disable services ▪ Deallocate runtime services ▪ Change allocation of services to servers ▪ Run services at a lower consumption mode ▪ Allocate/deallocate servers ▪ Change levels of service ▪ Change scheduling
Response measure	The measures revolve around the amount of energy saved or consumed and the effects on other functions or quality attributes.	Energy managed or saved in terms of: ▪ Maximum/average kilowatt load on the system ▪ Average/total amount of energy saved ▪ Total kilowatt hours used ▪ Time period during which the system must stay powered on . . . while still maintaining a required level of functionality and acceptable levels of other quality attributes

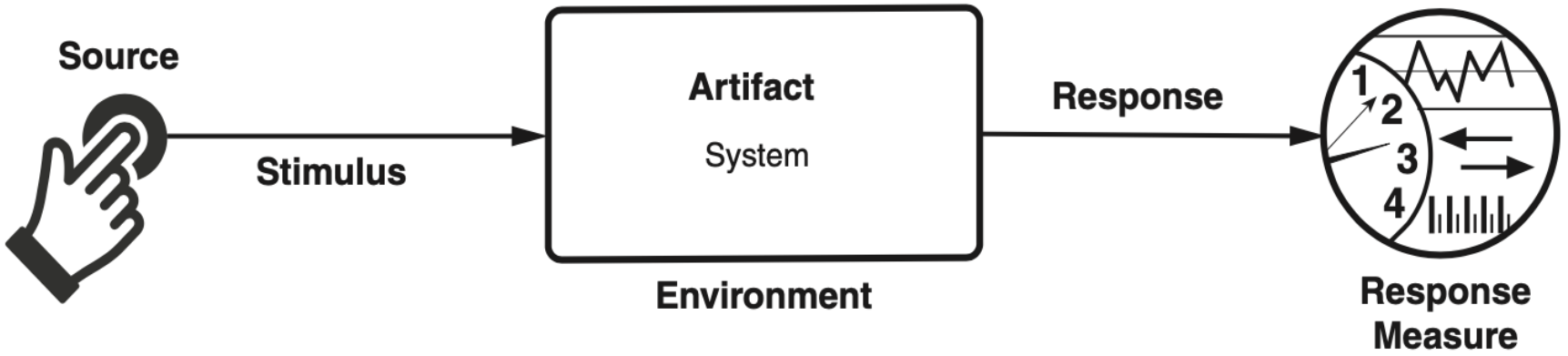


Sample Concrete Energy Efficiency Scenario

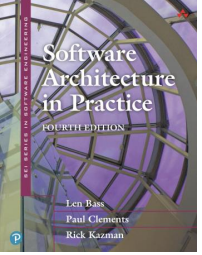
- *A manager wants to save energy at runtime by deallocating unused resources at non-peak periods. The system deallocates resources while maintaining worst-case latency of 2 seconds on database queries, saving on average 50 percent of the total energy required.*



Sample Concrete Energy Efficiency Scenario

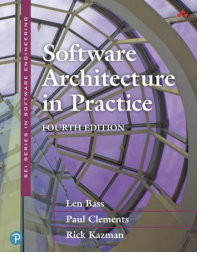


Source	Stimulus	Environment	Response	Response Measure
Manager	Wants to save energy at runtime by deallocating unused resources at non-peak periods	Non-peak period	The system deallocates resources	Maintaining worst-case latency of 2 seconds on database queries, saving on average 50% of the total energy required

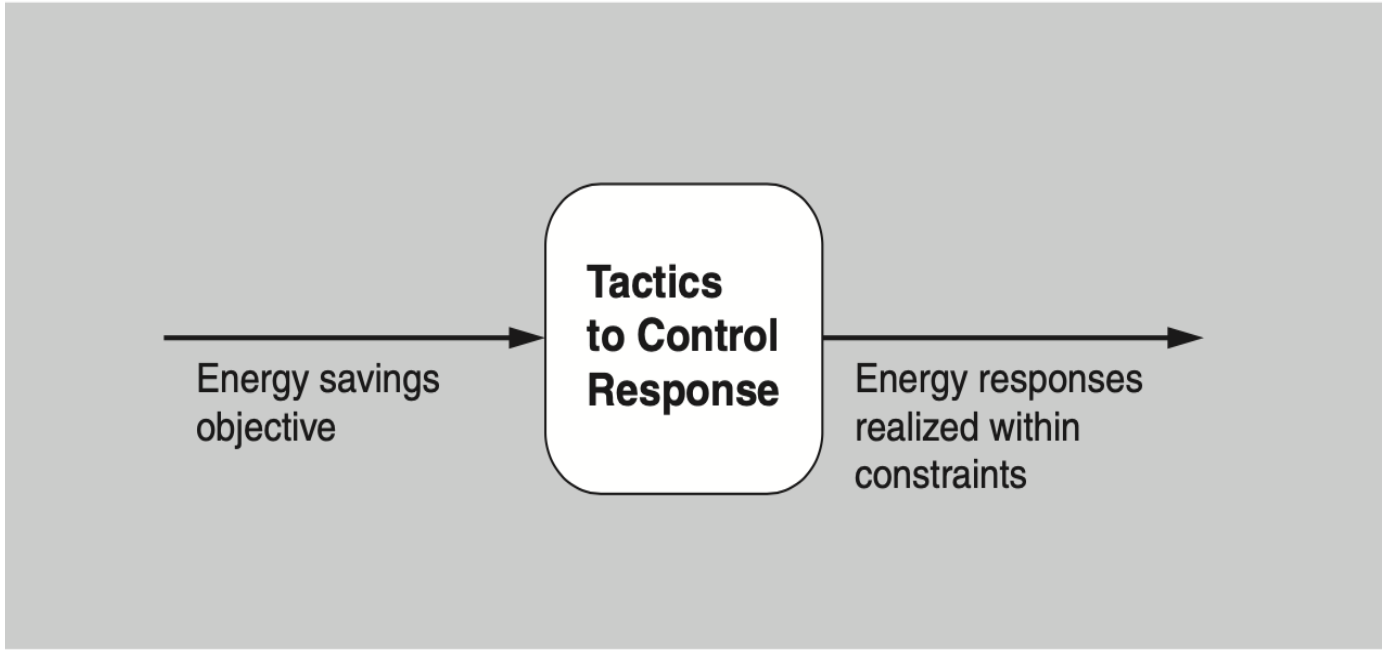


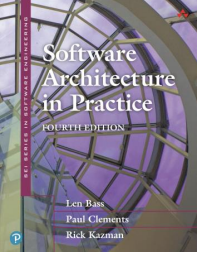
Goal of Energy Efficiency Tactics

- An energy efficiency scenario is catalyzed by the desire to conserve or manage energy while still providing the required functionality.
- This scenario is successful if the energy responses are achieved within acceptable time, cost, and quality constraints.

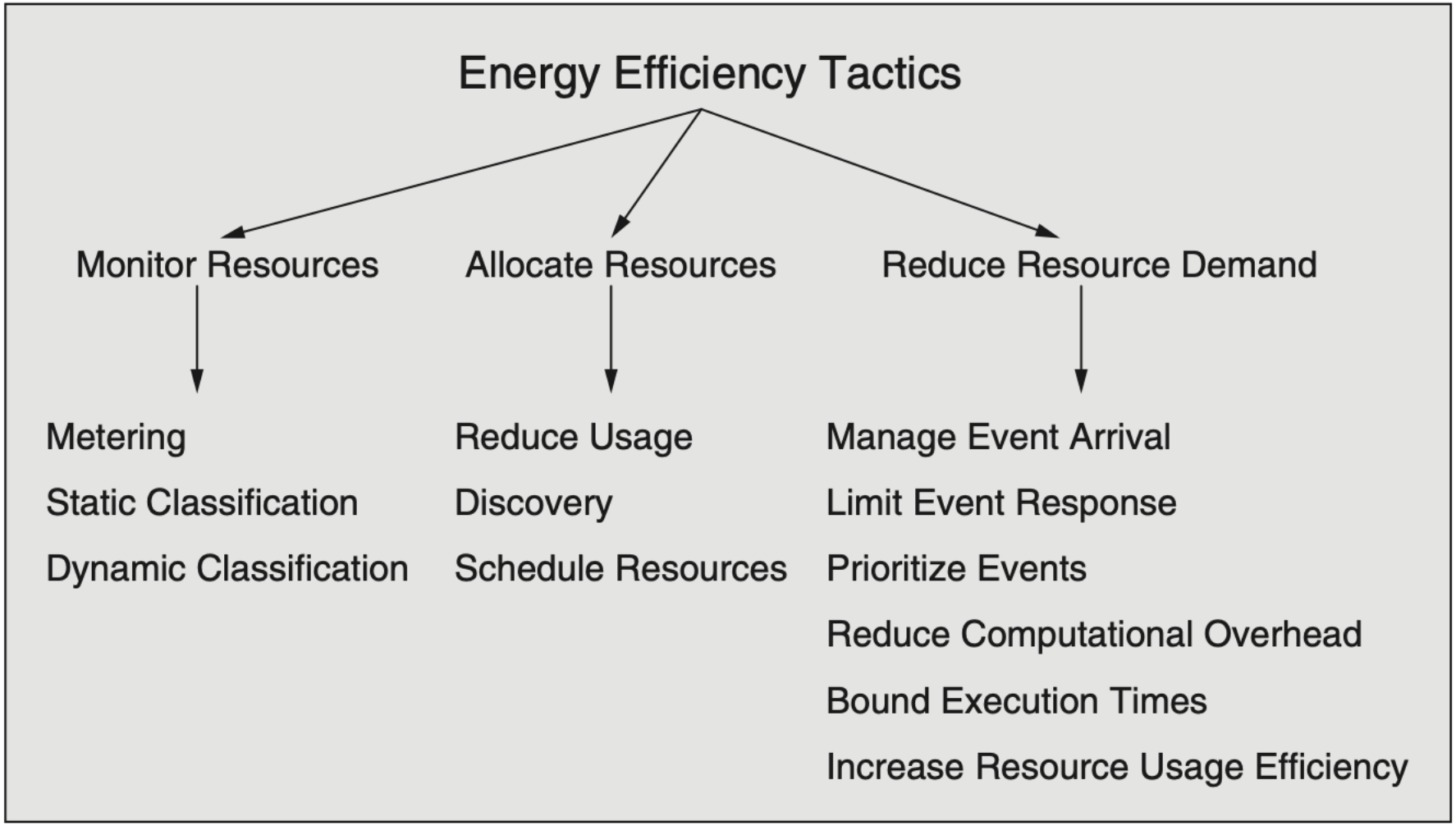


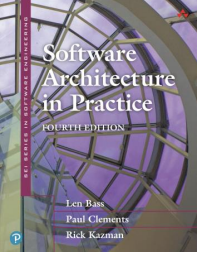
Goal of Energy Efficiency Tactics





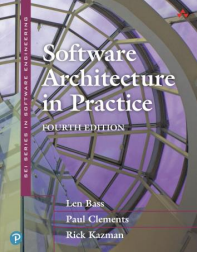
Energy Efficiency Tactics





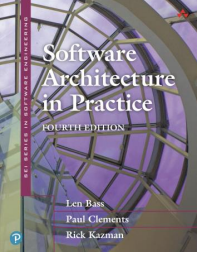
Monitor Resources

- *Metering*. The metering tactic involves collecting data about the energy consumption of computational resources via a sensor infrastructure, in near real time.
- *Static classification*. Sometimes real-time data collection is infeasible. Static classification allows us to estimate energy consumption by cataloging the computing resources used and their known energy characteristics. These characteristics are available as benchmarks, or from manufacturers' specifications.
- *Dynamic classification*. In cases where a static model of a computational resource is inadequate, a dynamic model might be required. Unlike static models, dynamic models estimate energy consumption based on knowledge of transient conditions such as workload.



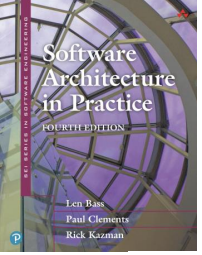
Allocate Resources

- *Reduce usage*. Usage can be reduced at the device level by device-specific activities such as reducing the refresh rate of a display or darkening the background. Removing or deactivating resources when demands no longer require them is another method for decreasing energy consumption.
- *Discovery*. As we will see in our discussion of Integrability (Chapter 7), a discovery service matches service requests with service providers. In the context of energy efficiency, this request could be annotated with energy information, allowing the requestor to choose a service provider based on its energy characteristics.
- *Schedule resources*. Scheduling is the allocation of tasks to computational resources. As we will see in our discussion of Performance (Chapter 9), the schedule resources tactic can increase performance. For energy, it can be used to manage energy usage, given task constraints and respecting task priorities.



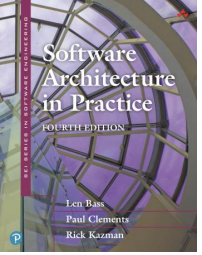
Reduce Resource Demand

- This category of tactics is taken from the tactics characterization of Performance (Chapter 9).



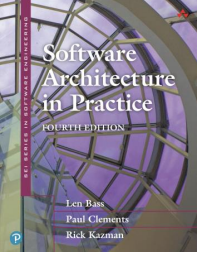
Tactics-Based Questionnaire for Energy Efficiency

Tactics Group	Tactics Question	Supported? (Y/N)	Risk	Design Decisions and Location	Rationale and Assumptions
Resource Monitoring	Does your system **meter the use of energy**? That is, does the system collect data about the actual energy consumption of computational devices via a sensor infrastructure, in near real time?				
	Does the system **statically classify** devices and computational resources? That is, does the system have reference values to estimate the energy consumption of a device or resource (in cases where real-time metering is infeasible or too computationally expensive)?				
	Does the system **dynamically classify** devices and computational resources? In cases where static classification is not accurate due to varying load or environmental conditions, does the system use dynamic models, based on prior data collected, to estimate the varying energy consumption of a device or resource at runtime?				



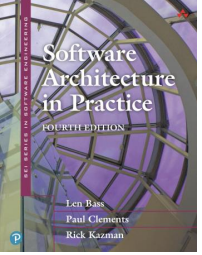
Tactics-Based Questionnaire for Energy Efficiency

Resource Allocation	Does the system **reduce usage** to scale down resource usage? That is, can the system deactivate resources when demands no longer require them, in an effort to save energy? This may involve spinning down hard drives, darkening displays, turning off CPUs or servers, running CPUs at a slower clock rate, or shutting down memory blocks of the processor that are not being used.
	Does the system **schedule resources** to more effectively utilize energy, given task constraints and respecting task priorities, by switching computational resources, such as service providers, to the ones that offer better energy efficiency or lower energy costs? Is scheduling based on data collected (using one or more resource monitoring tactics) about the state of the system?
	Does the system make use of a **discovery service** to match service requests to service providers? In the context of energy efficiency, a service request could be annotated with energy requirement information, allowing the requestor to choose a service provider based on its (possibly dynamic) energy characteristics.



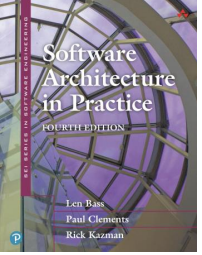
Tactics-Based Questionnaire for Energy Efficiency

Reduce Resource Demand	Do you consistently attempt to **reduce resource demand**? Here, you may insert the questions in this category from the Tactics-Based Questionnaire for Performance from Chapter 9.



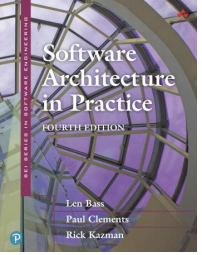
Sensor Fusion Pattern for Energy Efficiency

- Mobile apps and IoT systems often collect data from their environment using multiple sensors. In this pattern, data from low-power sensors can be used to infer whether data needs to be collected from higher-power sensors.
- A common example in the mobile phone context is using accelerometer data to assess if the user has moved and, if so, to update the GPS location.



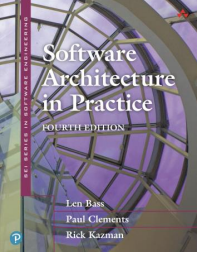
Sensor Fusion Pattern Benefits

- Benefits:
  - The obvious benefit of this pattern is the ability to minimize the usage of more energy-intensive devices in an intelligent way rather than, for example, just reducing the frequency of sampling.



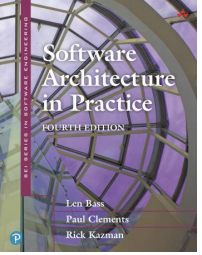
Sensor Fusion Pattern Tradeoffs

- Tradeoffs:
  - Consulting and comparing multiple sensors adds up-front complexity.
  - The higher-energy-consuming sensor will provide higher-quality data, albeit at the cost of increased power consumption. And it will provide this data more quickly, since using the more energy-intensive sensor alone takes less time than first consulting a secondary sensor.
  - If the inference frequently results in accessing the higher-power sensor, this pattern could result in overall higher energy usage.



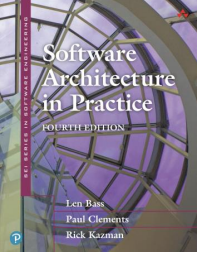
Kill Abnormal Tasks Pattern for Energy Efficiency

- Mobile systems, because they are often executing apps of unknown provenance, may end up unknowingly running some exceptionally power-hungry apps.
- This pattern provides a way to monitor the energy usage of such apps and to interrupt or kill energy-greedy operations.



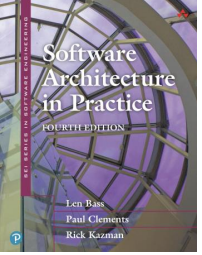
Kill Abnormal Tasks Pattern Benefits

- This pattern provides a “fail-safe” option for managing the energy consumption of apps with unknown energy properties.



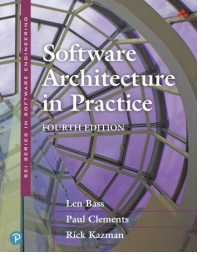
Kill Abnormal Tasks Pattern Tradeoffs

- Any monitoring process adds a small amount of overhead to system operations, which may affect performance and, to a small extent, energy usage.
- The usability of this pattern needs to be considered. Killing energy-hungry tasks may be counter to the user's intention.



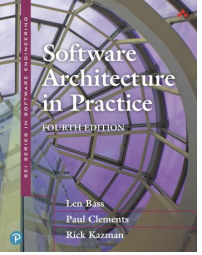
Power Monitor Pattern for Energy Efficiency

- The power monitor pattern monitors and manages system devices, minimizing the time during which they are active.
- This pattern attempts to automatically disable devices and interfaces that are not being actively used by the application.



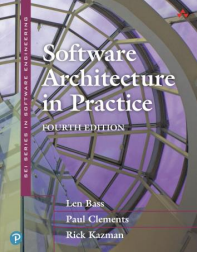
Power Monitor Pattern Benefits

- This pattern can allow for intelligent savings of power at little to no impact to the end user, assuming that the devices being shut down are truly not needed.



Power Monitor Pattern Tradeoffs

- Once a device has been switched off, switching it on adds some latency before it can respond, as compared with keeping it continually running. And, in some cases, the startup may be more energy expensive than a certain period of steady-state operation.
- The power monitor needs to have knowledge of each device and its energy consumption characteristics, which adds up-front complexity to the design.



Summary

- *Energy Efficiency* is a relatively new QA for architects to manage.
- But it can be managed in exactly the same way that we manage any other QA—by properly specifying energy scenarios, by paying attention to energy properties, and by designing with patterns and tactics.