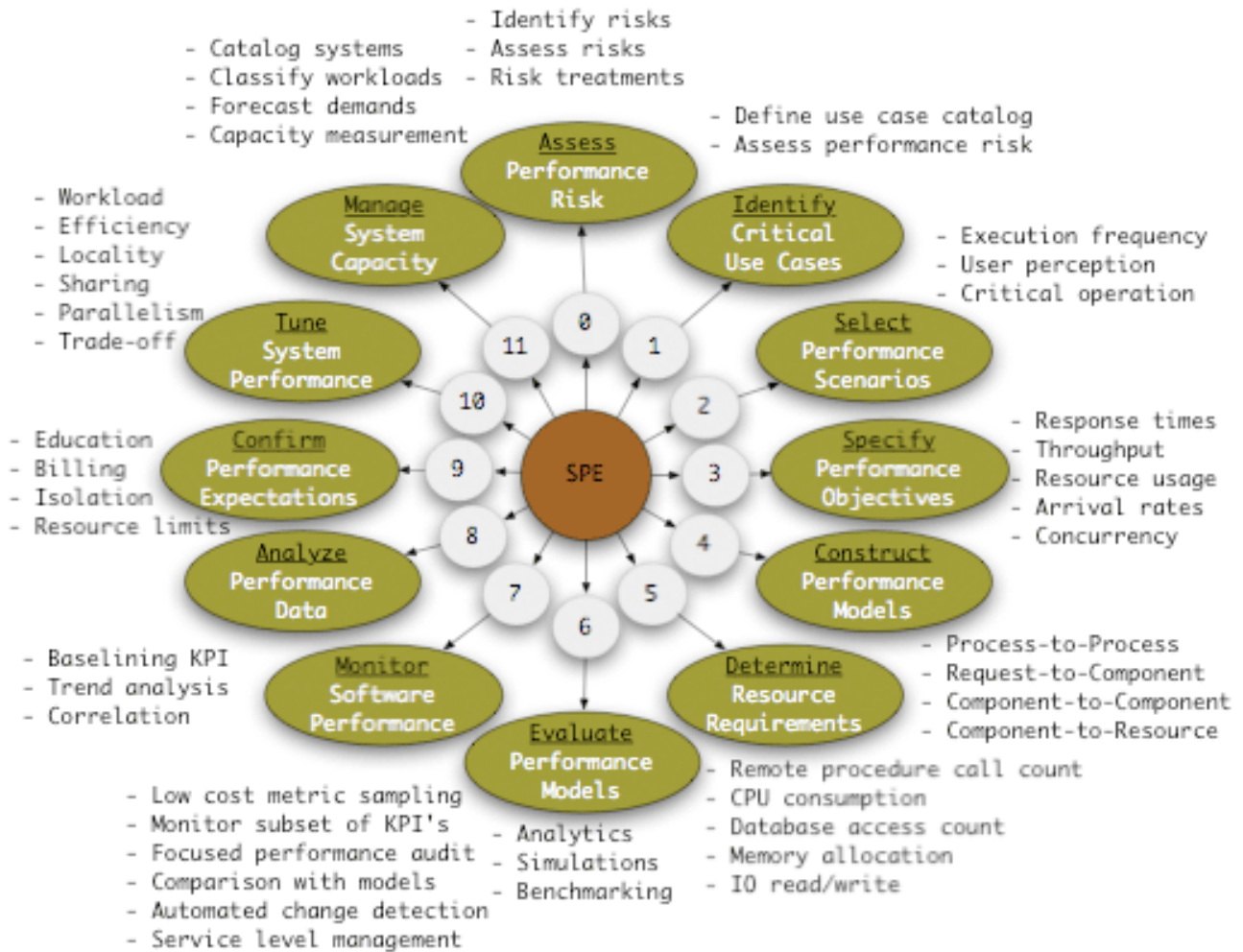
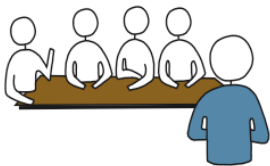


# JXINSIGHT SPE

**JXInsight Software Performance Engineer (SPE)** is pro-active performance assurance process spanning the complete software life cycle. It introduces a qualitative and quantitative method for software performance design and management. It details effective data gathering and performance measurement techniques and describes performance-oriented design patterns.



JXInsight SPE consists of 12 major activities that manage and monitor the performance of a software application or system from inception all the way through to production and across each release cycle.



## Access Performance Risk

During this activity risks are identified and assessed in terms of impact severity and probability with possible risk reduction treatments prescribed.

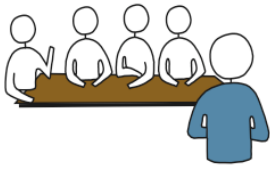
## Identify Use Cases

During this activity a use case catalog is created containing those use cases deemed to have performance risk and critical to the service delivery.



## Select Performance Scenarios

For each use case cataloged in the previous activity several performance scenarios are described in terms of execution frequency, user perception and criticality of operation.



### **Specify Performance Objectives**

During this activity performance objectives for each use case are defined. With each performance objective defined based upon whether it is realistic, reasonable, quantifiable, and measurable. Ideally performance objectives should balance the goals of the individual, community and enterprise.

### **Construct Performance Models**

During this activity both software and system execution models are constructed for each use case. Models are defined in terms of processing steps and interaction styles.



### **Determine Resource Needs**

During this activity resource requirements are ascertained for each of the performance scenarios per use case, detailing key performance indicators such as remote procedure call count, cpu consumption, memory allocation, and IO reads and writes.



### **Evaluate Performance Models**

During this activity various standard techniques such as benchmarking and simulation are applied in evaluating the performance models previously constructed factoring in variables including deployment topology, peak volumes, and arrival rates.

### **Monitor Software Performance**

During this activity various levels of monitoring are applied during the execution of the software. The level of monitoring can range from low cost metric sampling to focused performance audits. Additional information is also collected related to component and system dependencies.

### **Analyze Performance Data**

During this activity the monitoring data collected in the previous activity is analyzed. The analysis involves the determination of appropriate baselines for key performance indicators together with trend, cause and effect analysis.

### **Confirming Performance Expectations**

During this activity performance expectations are assessed involving various stakeholders. This can involve a degree of user education in terms of software usage patterns. For improved performance management the introduction of billing, partitioning and resource limits maybe warranted.

### **Tune Software and System Performance**

During this activity tuning is applied at various levels in the software and system stack with the goal to reducing performance bottlenecks via improved efficiencies and possible additional resource capacity.



### **Manage Capacity**

During this activity both the software and system are managed in terms of cataloging, classifications of workloads, demand forecasting, optimization and overhead reduction.