












Lightweight Project “Survival Kit”

Use the project flowchart below as a general guide for completing your projects.

	Activity	Project “Survival Kit” Handouts
1.0 	Start the project Receive assignment or assess a need to create a project charter	<ul style="list-style-type: none"> ■ Project Concept Template* ■ Project Charter Template* ■ Combo Meeting Toolkit* ■ Project Issues List*
2.0 	Develop a project plan Define scope & audience; identify requirements, deliverables, etc.	<ul style="list-style-type: none"> ■ Business & Technical Requirements* ■ Project Planning Checklist ■ <i>(Project Concept Template*)</i> ■ <i>(Project Charter Template*)</i> ■ <i>(Combo Meeting Toolkit*)</i>
3.0 	Form a project team Recruit needed talent; train team members; set up communication	<ul style="list-style-type: none"> ■ <i>(Project Charter Template*)</i> ■ <i>(Combo Meeting Toolkit*)</i> ■ <i>(Project Issues List*)</i>
4.0 	Estimate project needs Estimate the project task hours, resources, schedule, and budget	<ul style="list-style-type: none"> ■ Six Common Estimating Challenges ■ 12 Estimating Best Practices ■ Tips on Using Estimating Techniques ■ An Easy Project Scheduling Technique
5.0 	Assess the risks Identify and assess the risks; create a risk management plan	<ul style="list-style-type: none"> ■ Identifying Project Risks ■ Assessing and Mitigating Project Risks ■ <i>(Combo Meeting Toolkit*)</i> ■ <i>(Project Issues List*)</i>
6.0 	Design the deliverables Design and evaluate various alternatives for the deliverables	<ul style="list-style-type: none"> ■ <i>(Combo Meeting Toolkit*)</i> ■ <i>(Project Issues List)</i>
7.0 	Implement the solution Develop prototypes, implement, and/or roll out a pilot version	<ul style="list-style-type: none"> ■ <i>(Combo Meeting Toolkit*)</i> ■ <i>(Project Issues List*)</i>
8.0 	Test, revise, and evaluate Depending on deliverables, use various test & evaluation methods	<ul style="list-style-type: none"> ■ <i>(Combo Meeting Toolkit*)</i> ■ <i>(Project Issues List*)</i>
9.0 	Deliver & finalize the project Release and hand over to others; maintain; continuously improve	<ul style="list-style-type: none"> ■ Proactive Project Rollout Strategies ■ <i>(Combo Meeting Toolkit*)</i> ■ <i>(Project Issues List*)</i>

**Documents provided courtesy of the San Luis Obispo County Project Management Office as a separate set of handouts*



Project Planning Checklist

A quick-reference discussion tool to use with your team

<input type="checkbox"/>	1. How would you describe this project?
<input type="checkbox"/>	2. What are the goals and objectives?
<input type="checkbox"/>	3. Who will benefit from the project?
<input type="checkbox"/>	4. Will you be creating any products?
<input type="checkbox"/>	5. Will you be providing any services?
<input type="checkbox"/>	6. What general methods or approaches will you use?
<input type="checkbox"/>	7. What kind of phases, milestones, schedule, and budget do you anticipate?
<input type="checkbox"/>	8. Will you need any partners or collaborators?
<input type="checkbox"/>	9. Will you need specific information or expertise?
<input type="checkbox"/>	10. Will you need special systems, equipment, or resources?
<input type="checkbox"/>	11. Will you need to use special tools, templates, or conventions?
<input type="checkbox"/>	12. How will you evaluate project success?
<input type="checkbox"/>	13. Who needs to review and approve decisions?
<input type="checkbox"/>	14. How might your project evolve? How will you manage any derivatives later?
<input type="checkbox"/>	15. Who will be responsible for what?
<input type="checkbox"/>	16. What risks should you plan to manage?
<input type="checkbox"/>	17. What open questions remain?



Six Common Estimating Challenges

Studies of failed projects have revealed how difficult it can be to detect all the estimating red flags in advance. Unbridled optimism can block everyone’s ability to see clearly. Yet turning down an “iffy” project could be preferable to letting egos rule and allowing bravado to dominate the decision-making.

To what kinds of **estimating challenges** should we be paying more attention?

<input type="checkbox"/>	<p>1. Each project differs in some way, shape, or form from the last one. Even though projects may share some similarities, a new project could very easily introduce several new, unfamiliar elements that can completely throw off your sense of balance — often without your even realizing it until it’s too late.</p>
<input type="checkbox"/>	<p>2. People tend to underestimate (or overestimate) their time and effort. Whether it’s because of a perceived social stigma or a cloudy crystal ball, we typically have a difficult time deriving realistic project estimates. Given the number of project unknowns, coming up with accurate predictions can be tricky.</p>
<input type="checkbox"/>	<p>3. Project requirements are typically very fuzzy at the start. Whatever the stakeholders say initially may sound as clear as a bell in some areas but very sketchy in others. Getting clarification on the fuzzy parts might entail many conversations with many people, and much more time than anybody ever imagined.</p>
<input type="checkbox"/>	<p>4. Requirements invariably shift over time. The minute after we’ve cemented the requirements with everyone’s agreement, “scope creep” begins. The project needs might expand, shrink, or morph into something altogether different! These situations arise because <i>the very act of creating something new</i> can produce a set of results that may exceed, or differ from, what people were capable of imagining at the start.</p>
<input type="checkbox"/>	<p>5. Nearly everything about the project is dynamic. Aside from the requirements changing, many other things can stop, start, or fluctuate during the project. Experienced people may leave and new people may come on board. Budgets could get chopped. The schedules might get slashed or — sometimes even worse — delayed, and on and on.</p>
<input type="checkbox"/>	<p>6. Yet, projects are often constrained by finite conditions. Although it’s very understandable to want the greatest value for the money, usually it’s possible to attain only certain goals within the project scope. If both the budget and schedule are fixed, the scope would have to limit the quality, constrain the features, or both.</p>



12 Estimating Best Practices

Combine these techniques as applicable to create a set of robust estimating practices.

<input type="checkbox"/>	1. Maintain an ongoing “actual hours” database of recorded time spent on each aspect of your projects. Use the data to help estimate future projects and identify the historically accurate “buffer time” adjustments needed to realistically perform the work.
<input type="checkbox"/>	2. Create and use planning documents , such as specifications and project plans . In the estimating stage, these are indispensable in helping to define the project scope.
<input type="checkbox"/>	3. Perform a detailed task analysis of the work to be performed. If you are not intimately familiar with the tasks you need to estimate, interview the people who are.
<input type="checkbox"/>	4. Use a “complexity factor” as a multiplier to gauge whether a pending project is more or less complex than a previous one. For example, Project B might be 200% more complex than prior Project A because there will be twice as many deliverables.
<input type="checkbox"/>	5. Use multiple methods to arrive at an estimate , such as a detailed task analysis, a complexity factor from comparing the effort with a prior project, and an adjustment buffer derived from your “actual hours” database. Look for the midpoint among them.
<input type="checkbox"/>	6. Document caveats, constraints, and assumptions in your estimates to bound the conditions under which your estimates would be meaningful. Any work performed outside of those constraints would be considered out of scope.
<input type="checkbox"/>	7. Propose adjusting <i>cost, schedule, quality or features</i> upward or downward if the proposed budget or schedule seems inadequate to do the work involved. Usually clients can realistically expect only two or three out of four factors on a typical project.
<input type="checkbox"/>	8. Consider more efficient ways to organize and perform the work if the project will have many constraints. One way is to dedicate specialists to performing certain tasks in an assembly line mode, such as testing, formatting documents, or populating templates.
<input type="checkbox"/>	9. Plan and estimate the project rollout from the very beginning so that it won’t become a chaotic scramble at the end. For example, during the estimating phase, you can propose using a pilot program or a phased implementation during the rollout.
<input type="checkbox"/>	10. Consider a phase-based project approach , especially in very nebulous situations. The first phase focuses on requirements gathering and estimating subsequent phases.
<input type="checkbox"/>	11. Prioritize the deliverables with your clients or stakeholders, right from the start, into “ must-have ” and “ nice-to-have ” categories. Then, in case of a schedule crunch, concentrate on first delivering the “must-have” items in the previously agreed order.
<input type="checkbox"/>	12. Refer to your lessons-learned database for “20:20 foresight” on new projects. Consistently include best work practices in estimates so they’re always accounted for.



Tips on Using Estimating Techniques

Use the guidelines below to help you construct estimates for the effort required to complete your project. Using more than one method helps increase accuracy!

1. Identify the work breakdown structure (WBS) for your project. A WBS is simply a list of common activities you would perform on a typical project. (See examples on the next page.) If you've been keeping historical records on how long it takes to do these tasks, it will greatly help future estimating!

2. Estimate several ways, if possible, and compare the results, such as:

- ❑ **An off-the-top-of-your-head, "wild guess"** that might initially be very low or high. You can "reality-check" wild guesses using realistic adjustments that come from your historical, "actual hours" database.
- ❑ **A bottom-up, detailed task analysis** of all tasks to be performed (sample tasks on the next page).
- ❑ **A unit-based average** derived from your "actual hours" database — if you have accumulated enough data to generate reliable averages. You'd simply multiply the **units** of work to be performed (such as a number of pages, lines of code, drawings, lessons, etc.) by the **productivity rate** (the average number of hours it's historically taken to complete a unit).
- ❑ **A top-down complexity factor**, where you can compare a new project with a previous one if the tasks are relatively similar. Always include an explanation when using this approach such as, "Project A will most likely be about 2.5 times more complex than our earlier Project B because we'll be creating six new modules and completely redesigning the navigation."



3. Use the midpoint among these calculations as your best estimate. It represents the average of the range of optimistic and pessimistic values.

4. Note the amount of variability in your range of estimates. If the range between the lowest and highest estimates is fairly large, add a cautionary note about your confidence level and how you arrived at your estimates.



Sample Work Breakdown Structure for Comparing Estimates:

Activities and Tasks	Detailed task analysis	Unit-based averaging	Complexity factor
Administrative			
Meetings and correspondence			
Estimating, budgeting, planning, scheduling			
Travel			
Requirements Analysis			
Gathering and analyzing requirements			
Developing specifications			
Design			
Designing and developing prototypes			
Evaluating and selecting design options			
Usability testing (initial)			
Implementation			
Fabrication			
Software coding			
Documentation development & revisions			
Web site development & revisions			
Test and Evaluation			
Alpha testing			
Beta testing			
Usability testing (final)			
Documentation & Web content reviews			
Customer conversion testing (Web site)			
Delivery			
Final approvals and buy-off			
Rollout, setup, and training			
Establishing support systems			
Analyzing next-round change requests			
Reviewing "lessons learned"			
Total hours:			



An Easy Project Scheduling Technique

Planning your project schedule involves identifying the types of **activities** and **milestones** you'll need to accommodate, and placing them into some kind of order. Ideally, you and your team will want to do this together. **Note that:**

Activities reflected on the schedule typically come from the *work breakdown structure* that you've already created. If you don't have a WBS, simply brainstorm all of the actions that should occur using this exercise. Activities might occur in a serial mode, or more likely, some will overlap and run in parallel.

Milestones that you might want to include could be contractual dates, events, meetings, reviews, the completion of deliverables or subcomponents, decision points, vendor deliveries, approval dates, project deliveries, and so on.

Planning a Schedule with Activities, Milestones, and Dates

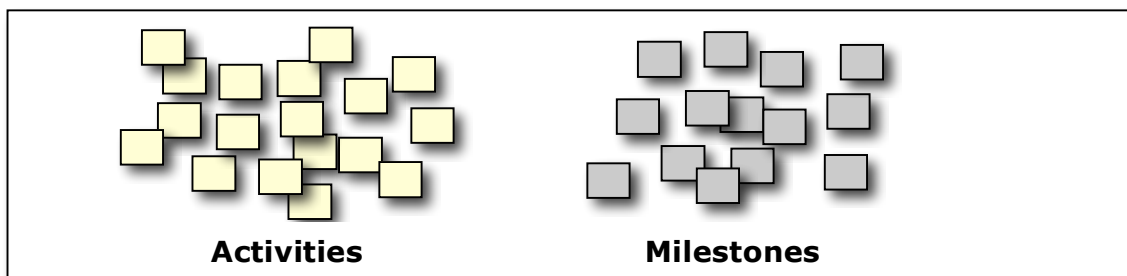
This is a fun, low-tech, hands-on way to get everyone's input on scheduling!



Exercise: Create a Rough-Cut Schedule (Part 1)

Instructions: Involve the team, management, and anyone else participating in the planning process. Tape a big piece of butcher paper (or flipchart sheets) to the wall.

- 1) Brainstorm individually or as a group: a) the **activities** to be performed, and b) the related **milestones**. (You could do this in two or more passes, if desired).
- 2) Record these ideas on **sticky notes** using two different colors, one for activities and the other for milestones, and place them randomly on the paper. An example of developing a schedule to prepare for a **trade show** appears on the next page.

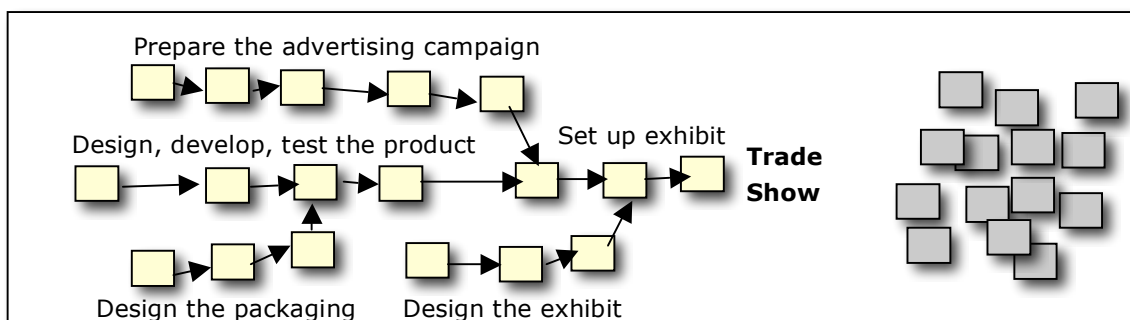




Exercise: Create a Rough-Cut Schedule (Part 2):

Instructions: As a group, gather around the sticky-note collection and:

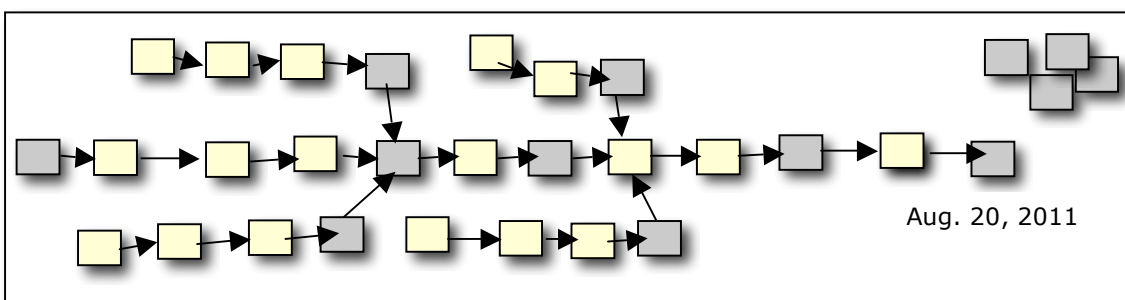
- 1) Begin to place **activities** in some kind of logical sequence. In the first pass, you and your team could do this quickly and silently. Keep looking for and filling in gaps in the activity sequence. Keep asking yourselves, "What are we missing?"
- 2) In subsequent passes, move the **activities** into more specific workflows, some of which will logically run in parallel with others. Determine where the activity **dependencies** exist in the flows, as shown in the **trade show** example below.



Exercise: Create a Rough-Cut Schedule (Part 3):

Instructions: As a group, continue working with the sticky-note collection, and:

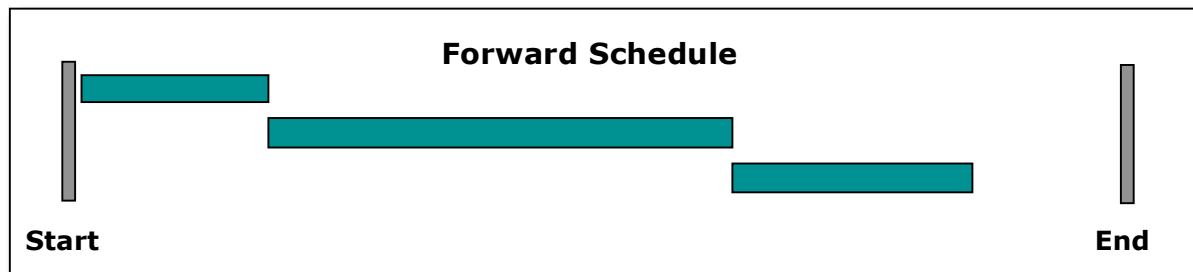
- 1) Begin to place **milestones** in appropriate places, refining the flow as needed.
- 2) In subsequent passes, start identifying all *fixed* and *flexible milestone dates*. For example, a trade show has a **fixed date** from which you'd work backward. Defining the project's *duration* should consider historical data, task estimates, and expert opinion; and potentially build in **contingency buffers** at strategic points to help mitigate lead-time risks (the subject of more advanced training).





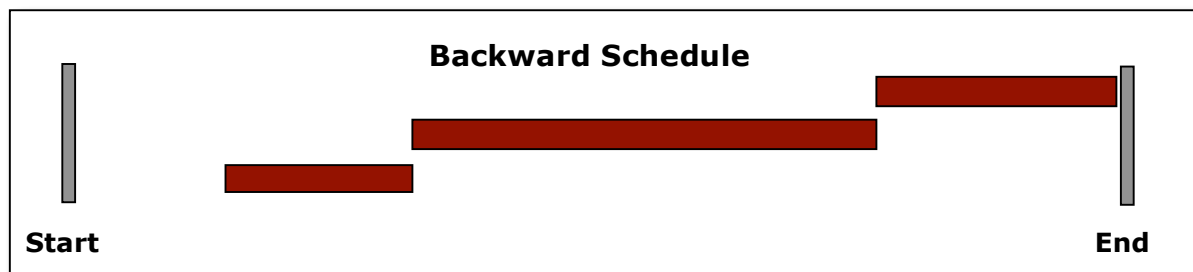
Consider the Schedule Both Forwards and Backwards

Note that that assigning milestone **dates** in the exercise above assumes that you are doing **forward scheduling** — which will give you some idea of how **soon** you can *start* various activities:



Adapted from SoftwareMatters.com, Inc., ©2005

In contrast, **backward scheduling** determines the **latest** time that you must start activities to avoid pushing out the completion date:



Adapted from SoftwareMatters.com, Inc., ©2005

With regard to **backward scheduling**, milestone events such as trade shows are good examples of fixed points that require reverse preparation. As shown in the prior examples, a show could be the strategic venue for a new product unveiling, a customer feedback opportunity, or a new marketing campaign.



In any case, it's helpful to look at forward and backward scheduling like two **oncoming trains** — at some point, an open-ended, *forward* schedule needs to brake and begin readying for the train speeding in from the opposite direction!



Identifying Project Risks

Use this checklist to think proactively about several kinds of risks your project could be facing.

<input type="checkbox"/>	<p>1. Is the project or solution non-compelling, or a bad fit for the team? A bad fit means it doesn't fall within the general professional or technical arenas in which your company or your colleagues have expertise.</p>
<input type="checkbox"/>	<p>2. Will the project scope entail operating in unfamiliar territory? Even if it's a reasonable fit, if a project involves requirements your team has never worked with before, can you ensure that everyone will come up to speed quickly enough to be successful on the project?</p>
<input type="checkbox"/>	<p>3. Are the project requirements and features vague or complex? Poorly defined requirements, or the need to implement many complicated features that must interact correctly, will vastly increase the potential for problems.</p>
<input type="checkbox"/>	<p>4. Are the requirements pitted against an aggressive schedule? Time limits of some sort exist on almost every project. Will there be enough time to implement the requested features at the desired quality level?</p>
<input type="checkbox"/>	<p>5. Are too few personnel and resources available for the project? Since it's easy to underestimate these needs, what are your fallback alternatives?</p>
<input type="checkbox"/>	<p>6. Will coordination with many different collaborators be needed? Any time you involve many people, it means complex hand-offs. If your project will include client or third party collaborators, how will people interact?</p>
<input type="checkbox"/>	<p>7. Are the primary collaborators unfamiliar to the project team? If and when it does become necessary to recruit one or more new contributors, how will you be able to verify whether they can do the job?</p>
<input type="checkbox"/>	<p>8. Are project team members discouraged from raising concerns? Do you want people to raise red flags when they see potential problems, or do you prefer everyone to maintain a stiff upper lip, and work 24/7 if needed?</p>
<input type="checkbox"/>	<p>9. Are there insufficient review and/or test cycles in the schedule? Is enough time allocated for review and testing iterations? If project delays add up, can you afford to reduce testing and still deliver a quality product?</p>
<input type="checkbox"/>	<p>10. Are there no standard protocols for managing scope changes? When the inevitable "add-on requests" materialize, think about how they'll affect the project. If unavoidable, will you have some kind of a method or system, such as a project change request, to adjust the official budget and schedule?</p>



Assessing and Mitigating Project Risks

Use a separate copy of this worksheet to score and mitigate each of the risks you have identified. (See http://learnshareprosper.com/tools/risk_management.pdf for an **interactive** version of this form.)

Name of Identified Risk: _____

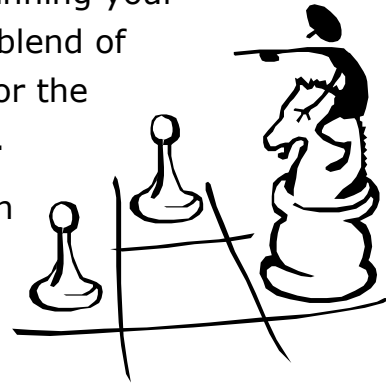
Assessment:					
1. What kinds of <i>problems</i> could this risk cause? How serious are they? How likely to occur?					
2. How might this risk negatively impact project <i>cost, schedule, quality, and features</i>? The higher the total score compared to other identified risks, the higher its relative priority.					
Enter an impact score for each area: Possible high impact=9, Medium impact=5, Low impact=1, No impact=0 ➔	Cost	Schedule	Quality	Features	TOTAL
Mitigation:					
3. Is there any way to completely <i>avoid</i> or <i>eliminate</i> this risk? If so, how?					
4. If you <i>can't avoid</i> or <i>remove</i> this risk, what are some risk management alternatives? (For example, are there ways of accepting, monitoring, alleviating, or transferring the risk?)					
5. Identify "do-nothing" and "next-to-nothing" alternatives. These are the options that would remain available if you <i>did not</i> attempt the project or solution. "Do nothing" means maintaining the status quo; "next-to-nothing" means using an option that already exists.					
6. For the four most logical alternatives (from #4 above), score each according to how well it seems to minimize negative impacts on cost, schedule, quality, and features . This helps determine whether you can successfully manage the risk, or should reconsider the project. You might use <i>one or more</i> high-scoring alternatives to minimize several potential impacts. High likelihood of minimizing an impact=9, Medium likelihood=5, Low likelihood=1, No likelihood=0					
Risk management alternative #1:	Cost	Schedule	Quality	Features	TOTAL
Risk management alternative #2:					
Risk management alternative #3:					
Risk management alternative #4:					



Proactive Project Rollout Strategies

The project rollout ideally represents a **planning opportunity** that occurs during the *estimating* stages of your project. By planning your moves strategically, you can aim to use whichever blend of options provides the greatest utility *and* flexibility for the project's participants, stakeholders, and customers.

Introducing these ideas **later** in the project also can work wonders, especially if it's becoming clear that there's too much complexity and not enough time to deploy everything by the scheduled due date.



- 1. Can you roll out your project as a pilot program?** Pilot rollouts offer a small-scale way to put the project to a real-world test with a smaller group of people, and to fix any problems before a large-scale deployment occurs. A *prototype* also lets people try out the look, feel, and initial functionality.
- 2. Can you complete and phase in the features over time?** If there are more features than anyone could possibly implement in the time available, you could propose initially delivering a *skeletal version* of the final product — one that has a “Cadillac” framework but only “Volkswagen” functionality. You could deliver the remaining features *just in time* for their needed uses.
- 3. Can you supply completed project modules in stages?** Similar to #2, you might offer to deliver a series of fully functional, standalone segments, one by one. That way, the customers could thoroughly evaluate each one under real-world conditions, and you can address any problems and make needed adjustments before rolling out the next segment.
- 4. Can you use “must-have” vs. “nice-to-have” delivery prioritization?** If you didn't do this at the beginning and you encounter a schedule crunch, you could propose delivering only the highest priority, “**must-have**” items that will be needed most frequently, or that would produce the most critical impacts if they were missing. Then plan to defer to a future project or phase the completion of all “**nice-to-have**” items that are not as time-sensitive.