AVOIDING THE PITFALLS OF CAPITALIZING SOFTWARE IN AN AGILE WORLD

Paul Argiry, CFO
WHY THE CLOUD?

Benefits of cloud apps that firms most often rate very important or important

- 68% Improved Business Agility
- 68% Lower Overall Cost
- 67% Speed of Implementation
- 66% Easier Information Sharing
- 63% Allows us to Focus Resources Elsewhere
- 59% Delivers Unique Functionality

By 2019 almost half of global technology purchases via non-IT buyers.

When the business is ready to move forward...
When the business is ready to move forward...

...Don’t forget to include Finance & Accounting...
...Or they could block you
WHAT WE’LL COVER TODAY

• Accounting for Software Development Costs
• Methods to Track Costs
• Table Exercise – Time vs. Story Points
• Features Completed Alternative
• Amortization
• Impact of the Cloud
• Cloud and Website Development Accounting
• “20% Limit”
• Capitalization Opportunity
SOFTWARE DEVELOPMENT

Accounting for and Capitalizing Costs
COMMON QUESTIONS

1. We’ve done capitalization under waterfall for decades – what happens under agile?

2. With the constant deployment of software and continual iterations everything is expensed, right?

3. How would we track these costs within an agile environment - must be different?
CAPITALIZATION
TYPES OF CAPITALIZATION

INTERNAL-USE SOFTWARE

ASC 350-40
SOP 98-1

SOFTWARE TO BE SOLD, LEASED OR MARKETED

ASC 985-20
FASB 86
# Development Stages
## Internal-Use Software

<table>
<thead>
<tr>
<th>Preliminary Project Stage</th>
<th>Application Development Stage</th>
<th>Post-Implementation Operation Stage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Costs expensed as incurred</td>
<td>Most costs capitalized as incurred</td>
<td>Costs expensed as incurred</td>
</tr>
<tr>
<td>Conceptual formulation of alternatives &amp; requirements</td>
<td>Design of chosen path, including software configuration and interfaces</td>
<td>Deployment</td>
</tr>
<tr>
<td>Evaluation of alternatives</td>
<td>Coding</td>
<td>Training</td>
</tr>
<tr>
<td>Determination of existence of needed technology</td>
<td>Installation of hardware</td>
<td>Application maintenance</td>
</tr>
<tr>
<td>Final selection of alternatives</td>
<td>Testing - including parallel processing phase</td>
<td>Support</td>
</tr>
</tbody>
</table>
CAPITALIZATION PERIOD
INTERNAL-USE SOFTWARE

BEGIN

- Preliminary Stage completed
- Management authorizes & commits to funding
- Probable project will be completed & software used as intended

END

- Computer software substantially complete
- Ready for intended use
- After all substantial testing completed
**CAPITALIZATION PERIOD**

**INTERNAL-USE SOFTWARE**

BEGIN

- Preliminary Stage completed
- **Management authorizes & commits to funding**
- Probable project will be completed & software used as intended

END

- Computer software substantially complete
- Ready for intended use
- After all substantial testing completed
WATERFALL ALIGNS NICELY WITH ACCOUNTING GUIDANCE
INTERNAL-USE SOFTWARE

Requirements
Analysis
Coding
Verification
Deployment
Maintenance

PRELIMINARY PROJECT
APPLICATION DEVELOPMENT
POST-IMPLEMENTATION OPERATION

Management commits to funding
Transitioning to agile is not problematic because:

- It is the nature of the cost, not entirely timing of their occurrence, that matters.
- These do not change under either software development model – waterfall or agile.
- Tracking of activities is key for capitalization.
- Preliminary Project Stage must be completed.
NATURE OF COSTS, NOT TIMING, IS MOST IMPORTANT WITHIN AGILE
INTERNAL-USE SOFTWARE

Management commits to funding

CONCEPTUAL FORMATION & EVALUATION

- PRELIMINARY PROJECT
  - Expense as incurred

DEPLOYMENT & MAINTENANCE

- APPLICATION DEVELOPMENT
  - Most costs capitalized as incurred

- POST-IMPLEMENTATION OPERATION
  - Expense as incurred

DESIGN AND CODING

- 2-week sprint
- 2-week sprint
- 2-week sprint
**EXPENSED ITEMS**

Internal-Use Software

<table>
<thead>
<tr>
<th>EXPENSE</th>
<th>STAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overhead &amp; administrative support</td>
<td>All Stages</td>
</tr>
<tr>
<td>Determine business, performance &amp; system requirements</td>
<td>Preliminary</td>
</tr>
<tr>
<td>Create business process &amp; IT solution proposal</td>
<td>Preliminary</td>
</tr>
<tr>
<td><strong>Business process re-engineering</strong></td>
<td>Preliminary</td>
</tr>
<tr>
<td>Create project proposal</td>
<td>Preliminary</td>
</tr>
<tr>
<td>Complete business case</td>
<td>Preliminary</td>
</tr>
<tr>
<td>Establish project team &amp; management</td>
<td>Preliminary</td>
</tr>
<tr>
<td>Strategic decisions allocating resources</td>
<td>Preliminary</td>
</tr>
<tr>
<td>Develop project schedule</td>
<td>Preliminary</td>
</tr>
<tr>
<td>Host project kick off meeting</td>
<td>Preliminary</td>
</tr>
<tr>
<td><strong>Create &amp; deliver user training</strong></td>
<td><strong>Dev &amp; Post-Implement</strong></td>
</tr>
<tr>
<td><strong>Perform data conversion</strong></td>
<td><strong>Development</strong></td>
</tr>
<tr>
<td>Deploy solution</td>
<td>Post-Implementation</td>
</tr>
<tr>
<td>Activate &amp; transition Go-Live support team</td>
<td>Post-Implementation</td>
</tr>
<tr>
<td>Monitor solution performance &amp; metrics</td>
<td>Post-Implementation</td>
</tr>
<tr>
<td>Maintenance</td>
<td>Post-Implementation</td>
</tr>
<tr>
<td><strong>Bug fixes</strong></td>
<td><strong>Post-Implementation</strong></td>
</tr>
<tr>
<td>Determine final project costs</td>
<td>Post-Implementation</td>
</tr>
</tbody>
</table>
**CAPITALIZED ITEMS**

Internal-Use Software

<table>
<thead>
<tr>
<th>CAPITALIZE</th>
<th>STAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software licenses including conversion software</td>
<td>Any stage</td>
</tr>
<tr>
<td>Design requirements and document processes</td>
<td>Development</td>
</tr>
<tr>
<td>Define interfaces, conversions &amp; enhancements</td>
<td>Development</td>
</tr>
<tr>
<td>Install technical development environment</td>
<td>Development</td>
</tr>
<tr>
<td>Test performance</td>
<td>Development</td>
</tr>
<tr>
<td>Conduct readiness review</td>
<td>Development</td>
</tr>
<tr>
<td>Finalize end-to-end test plan</td>
<td>Development</td>
</tr>
<tr>
<td>Coding &amp; Development</td>
<td>Development</td>
</tr>
<tr>
<td>Install hardware</td>
<td>Development</td>
</tr>
<tr>
<td><strong>Sprint Planning</strong></td>
<td>Development</td>
</tr>
<tr>
<td><strong>Retrospectives</strong></td>
<td>Development</td>
</tr>
<tr>
<td>Complete user-acceptance testing</td>
<td>Development</td>
</tr>
<tr>
<td>Perform stress testing</td>
<td>Development</td>
</tr>
<tr>
<td>Complete final deployment plan</td>
<td>Development</td>
</tr>
<tr>
<td><strong>Additional functionality – enhancements/upgrade</strong></td>
<td>Development</td>
</tr>
<tr>
<td><strong>Travel for consultants &amp; internal employees</strong></td>
<td>Development</td>
</tr>
</tbody>
</table>
CAPITALIZATION PERIOD
SOFTWARE TO BE SOLD, LEASED OR MARKETED

Expense as R&D
- Planning
- Designing
- Coding
- Testing

Capitalize: Costs must be recoverable
- Overhead is capitalizable
- G&A still expensed

Expense all subsequent costs
- Maintenance
- Customer Support
AVOID THIS PITFALL WITHIN TECHNOLOGICAL FEASIBILITY

Ensure **High-Risk** development issues:

- Novel
- Unique
- Unproven functions/features

Have been resolved via coding and testing before capitalizing costs
METHODS TO TRACK COSTS
METHODS TO TRACK COSTS FOR CAPITALIZATION

**TIME**
Each team member submits time into a time tracking system

**STORY POINTS**
Scrum story points are used to capitalize costs incurred by the entire team
TABLE GROUP EXERCISE

Time vs. Story Points
# TRACKING COSTS PROS & CONS

<table>
<thead>
<tr>
<th>PROS</th>
<th>STORY POINTS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TIME</strong></td>
<td><strong>STORY POINTS</strong></td>
</tr>
<tr>
<td>Align with current practices</td>
<td>Potentially higher correlation with team’s effort → so more stories/features completed = higher capitalization</td>
</tr>
<tr>
<td>Should be intuitive for employees to understand</td>
<td>Data naturally captured within scrum</td>
</tr>
<tr>
<td><strong>CONS</strong></td>
<td><strong>CONS</strong></td>
</tr>
<tr>
<td>Impacts productivity if done accurately → counter to agile &amp; scrum</td>
<td>Story points are subjective for each team so aligning consistency may prove challenging</td>
</tr>
<tr>
<td>Inaccurate if done fast to avoid impacting productivity</td>
<td>Overall use of story points based on estimation which auditors may frown upon</td>
</tr>
</tbody>
</table>
39% Idle & Admin time

Source: 2013 Electric Cloud Survey of 1,146 respondents
FEATURES
COMPLETED
ALTERNATIVE
FEATURES COMPLETED

- Two Teams – Team A and Team B
- 6 Team Members per Team
- $100,000 per person for Salary & Benefits
- $600,000 Total Salary & Benefits cost (i.e. $100,000 x 6)
- Four-month Release cycle
- 25% Idle and wait time = $150,000 (i.e. 1/4 of $600,000)
- Annual Capitalizable Base = $450,000 for each Team (i.e. $600,000 total - $150,000 idle time)
During Release, Team A completes:
- New Functionality Features (capitalizable) 320 points
- Bug Fixes (expensed) 240 points
- Small Enhancements (expensed) 240 points
Total 800 points

New Functionality Features represent 40% (i.e. 320/800) of work completed by Team A

$450,000 Annual Capitalizable Base - so 4 month Release is $150,000 (i.e. one third of $450,000)

Capitalize $60,000 ($150,000 x 40%) as software development costs for Team A
During Release, Team B completes:
- New Functionality Features (capitalizable) 50 points
- Bug Fixes (expensed) 25 points
- Small Enhancements (expensed) 25 points
Total 100 points

New Functionality Features represent 50% (i.e. 50/100) of work completed by Team B

$450,000 Annual Capitalizable Base - so 4 month Release is $150,000

Capitalize $75,000 ($150,000 x 50%) as software development costs for Team B
COMPARISON

IF GREAT DISPARITY IN STORY POINTS AMONG TEAMS, USING FEATURES COMPLETED IS RECOMMENDED COURSE OF ACTION

Team A
- Total Story Points completed 800
- 320 Points related to Features
- Features completed of total work = 40%
- $60,000 of capitalizable software development costs

Team B
- Total Story Points completed 100
- 50 Points related to Features
- Features completed of total work = 50%
- $75,000 of capitalizable software development costs
AMORTIZATION
INTERNAL-USE SOFTWARE

CAPITALIZATION BEGINS

Preliminary Stage completed
Management authorizes & commits to funding
Probable project will be completed & software used as intended

• Computer software substantially complete
• Ready for intended use
• After all substantial testing completed

CAPITALIZATION ENDS

AMORTIZATION PERIOD

Amortization begins when:
• Software is ready for intended use
• All substantial testing completed
• Amortization begins regardless software placed in service in planned stages
• For dependent modules needed to function, amortization begins when both that module and dependent modules are ready for their intended use
• Amortized over estimated useful life (e.g. 4 years)
AMORTIZATION PERIOD OPTIONS
INTERNAL-USE SOFTWARE

EACH RELEASE

• Capitalize software at each specific release and amortize immediately

• Given overall project has one defined useful life, subsequent capitalization and amortization over remaining useful life

• Assume 4-year useful life:
  • 1st release amortized over 48 months
  • 2nd release over 47 months
  • 3rd release over 46 months, etc.

EACH EPIC

• Software deployed in larger Epics (i.e. every 4-6 months)

• Capitalized and amortized based on that specific Epic and its related useful life of 4 years

• Subsequent Epic would start anew with its own useful life - could be 3, 4 or 5 years
AMORTIZATION
INTERNAL-USE SOFTWARE

CONCEPTUAL FORMATION & EVALUATION

DEPLOYMENT & MAINTENANCE

PRELIMINARY PROJECT

APPLICATION DEVELOPMENT

POST-IMPLEMENTATION OPERATION

DESIGN AND CODING

2-week sprint
2-week sprint
2-week sprint

Capitalization

Amortization Period
AMORTIZATION
SOFTWARE TO BE SOLD, LEASED OR MARKETED

• Amortized on a product by product basis
• Starts when product available for General Release
• Based on ratio of current software revenue compared to total expected revenue over software life
• Regular impairment tests → Write down to Net Realizable Value
## SOFTWARE DEVELOPMENT COSTS

<table>
<thead>
<tr>
<th></th>
<th><strong>INTERNAL USE</strong></th>
<th><strong>TO BE SOLD, LEASED OR MARKETED</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><em>(ASC 350-40)</em></td>
<td><em>(ASC 985-20)</em></td>
</tr>
<tr>
<td><strong>Begin Capitalization</strong></td>
<td>• Preliminary project stage complete&lt;br&gt;• Management authorizes and commits funding; and&lt;br&gt;• Probable project will be complete</td>
<td>• Technological feasibility reached; and&lt;br&gt;• Costs are recoverable</td>
</tr>
<tr>
<td><strong>End Capitalization</strong></td>
<td>• Testing substantially completed and software ready for intended use; or&lt;br&gt;• Unlikely project will be completed</td>
<td>• Available for General Release</td>
</tr>
<tr>
<td><strong>Amortization</strong></td>
<td>• Systematic over useful life</td>
<td>• Proportion of software generated revenue compared to total revenue expected over software life</td>
</tr>
</tbody>
</table>
IMPACT OF THE CLOUD
EXPANDING CLOUD WORKLOADS

DIGITAL TRANSFORMATION

26%  
MORE PROFITABLE

12%  
HIGHER MARKET VALUATION

Source: MIT Center for Digital Business
AVOID THE “UH-OH” AND FOCUS ON “A HA”

Digital Transformation Two Responses:

1 - Get ahead of Trends (“a ha”) @ 37%

2 - Respond to effects of Change (“uh oh”)

“TRANSFORM OR PERISH”

One Third

Top 20 Companies in Industry Segments

BUT WAIT... THERE'S MORE...
MORE ACCOUNTING FOR SOFTWARE DEVELOPMENT

CLOUD
ASU 2015-05

WEBSITE
ASC 350-50
THINK BEFORE YOU LEAP...

Within the Cloud - Software Capitalization:

- “Software To Be Sold” accounting **IF**:
  1. Take possession of software, **and**
  2. Run in another hosted environment - **without incurring a significant penalty**

- If not both = “Internal-Use Software”

**Plan Ahead** - “Trial & Error” of developing solution expensed if both conditions met under “Software To Be Sold” → otherwise capitalize costs under “Internal-Use”
# Website Development Costs

<table>
<thead>
<tr>
<th>Planning</th>
<th>Application &amp; Infrastructure Development</th>
<th>Graphics Development</th>
<th>Content Development</th>
<th>Operating</th>
</tr>
</thead>
<tbody>
<tr>
<td>All costs expensed as incurred</td>
<td>Capitalize</td>
<td>Capitalize &amp; Expense</td>
<td>Capitalize &amp; Expense</td>
<td>Capitalize &amp; Expense</td>
</tr>
</tbody>
</table>

### Expense

- Business Plan
- Competitive Analysis
- Hardware evaluation
- Web application eval

### Capitalize

- Acquire/develop development tools
- Obtain internet domain
- Acquire/develop website ops software, internet server, web browser
- Create hyperlinks
- Testing

<table>
<thead>
<tr>
<th>None</th>
<th>None</th>
<th>None</th>
<th>None</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

- Development of initial graphics
- Substantial enhancements to graphics
- Integrate website and database
- Upgrades & enhancements that add additional functionality → otherwise expense

- Ongoing maintenance of graphics
- Content into website Data conversion
- Train employees
- Administrative Maintenance
- Register website w/ search engines
IT Budgets should limit depreciation and amortization to 20% of total IT spend

Source: Gartner’s “Opex vs. Capex: CIOs Should Partner with CFOs” May 21, 2015
CAPITALIZATION

OPPORTUNITY
KEY CONSIDERATIONS

- Business Transformation
- Balance Mode 1 and Mode 2
- Change From Project Based to Product / Solution Based
- Reduce / Remove Impediments & Admin Burden

**Desired Result =**

**Increased Cap Rate & Lower Overall Dev Costs**
QUESTIONS?

LEADING AGILE