Sources of User Error in the XP Security Interface

Roy A. Maxion

Dependable Systems Laboratory
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213
Email: maxion@cs.cmu.edu

IFIP WG 10.4
03-07 July 2004
Siena, Italy

Acknowledgements

- Rob Reeder
- Fahd Arshad

Achtung!

- Note: This work is very preliminary.
- Goal is to begin understanding sources of undependability in terms of user error.
- Suggestions are welcome (but after my 30-minute slot, please). ☺

In search of a dependable user interface

- What makes interfaces undependable?
  - They don’t meet their specification.
- Specification (arbitrary, but could be based on cost of error)
  - 95% of users meet speed and accuracy targets
- Accuracy
  - The correct result must be achieved.
- Speed (arbitrary, based on multiple of expert time, cost, etc.)
  - 3x expert time (3 x 30 = 90 sec)

Motivation

- General frustration with user interfaces
- “Why Johnny Can’t Encrypt”
- Observations about XP in particular
- “Memogate”
- Etc.

Why Johnny Can't Encrypt (i)

Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0

- User study of PGP (pretty good privacy) email
- Indicated that users would commit errors such as sending someone a user’s private key

- **Why the paper was good**
  - was a seminal work in the field of usability for security interfaces
  - one of the first papers to make security researchers and designers aware that usability is as important as good algorithms in ensuring security

Why Johnny Can't Encrypt (ii)

Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0

- **Why the paper wasn't satisfying**
  - describes a lot of what users got wrong, but not why
  - no explanations to help designers improve
  - doesn't explain what aspects of the interface led people astray
  - starts with a cognitive walkthrough of the PGP 5.0 email encryption application, then gives user-test results, but never says whether experimental results confirm cognitive walkthrough results
  - offers design guidelines, but doesn't quantify them (e.g., "Security software is usable if the people who are expected to use it ... are reliably made aware of the security tasks they need to perform" - but "reliably" is unclear.

Other evidence re: XP security interface problems (i)

- Memogate
  - Republican staffers on Capitol Hill stole confidential Democratic memos from a shared Windows server because the system administrator had not set file permissions correctly.

    U.S. Senate Sergeant at Arms. *Report on the Investigation into Improper Access to the Senate Judiciary Committee's Computer System*. March, 2004.

Other evidence re: XP security interface problems (ii)

- At CMU
  - Our local Windows network administrator reports that many users make private files world-readable, because it’s too confusing to set permissions as desired.
  - That is, many users know what they ought to do, but can’t figure out how to achieve it.

Other evidence re: XP security interface problems (iii)

- Etc.
  - A Web search on the Experts Exchange tech advice site shows that there are many questions on Windows permissions rules, terminology, and capabilities.
  - Microsoft publishes a list of "best practices" that advises users not to use several of the features of the XP security interface, such as Deny-permissions and the ability to set permissions on individual files (as opposed to folders).

http://www.microsoft.com/mspress/books/sampchap/6103e.asp

So … we did a pilot study

- TEST:
  - We asked 24 users (mostly CMU PhD-level computer science majors) to manipulate security settings in various ways, using the XP interface.

- RESULTS:
  - Accuracy: only 62% completed task accurately, and these subjects did not check their results.
  - Speed: only 40% of accurate users completed task within 3x of expert speed; 60% within 5x.

Why these (seemingly serious) problems?

- 210 million copies of XP sold.

  (www.ciol.com/content/news/2004/104050307.asp - 03 May 2004)

- XP interface would seem to be designed for the average sort of user …for whom it is also important to get security right; security isn’t just for experts.

- Expectations of user skills probably not extremely high (3-5x not high expectation).

- So … what could be wrong?

Hypothesis: problems due to user error

- “Obviously,” users are committing errors … acts that fail to achieve what should have been done.
- So, we decided to explore/ask …
  - What kinds of errors do users make?
  - How many?
  - What are the causes of these errors?
  - What can be done about them?

Focus on one subtask of study - Ginny

- Why this subtask? Simple, short & representative.

- General methodology
  - Define security-related task
  - Run subjects
  - Log digital data
    - Mouse
    - Keyboard
    - Verbalization
    - Video
  - Analyze data (detect instances of user error)
  - Classify errors into one of four types
  - Look for causes/explanations of error
  - Suggest remedy

Ginny task

- Ginny (username: ginny) tried to access the file projectAdata.txt, but could not do so.

- She asks that she be allowed to read projectAdata.txt.

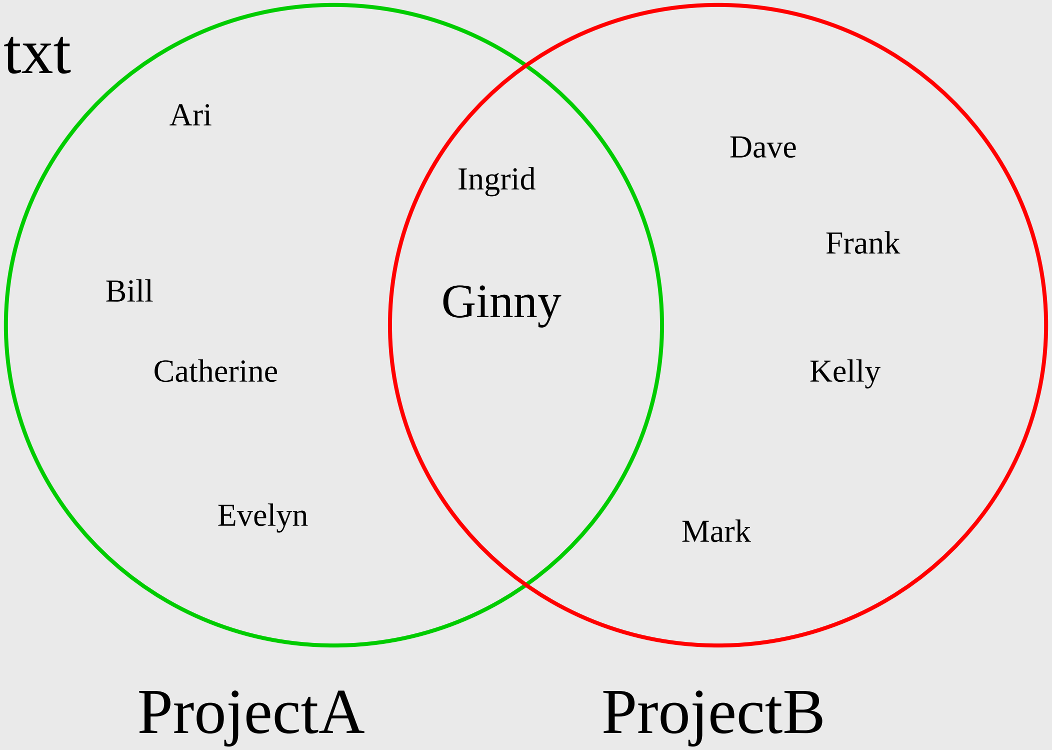
- TASK: Set permissions on projectAdata.txt so that Ginny can read the file.

Requirements for task solution

- User Goals: achieve end state
  - Method A – Remove Deny-Read from ProjectB
  - Method B – Give Allow-Read to ProjectB
- User Plans: procedure for moving from initial state to end (goal) state
- User Actions: sensorimotor faculties
- User Perceptions: correct perception and interpretation of syntax and semantics of events or objects presented in the user interface

Ginny task – initial state

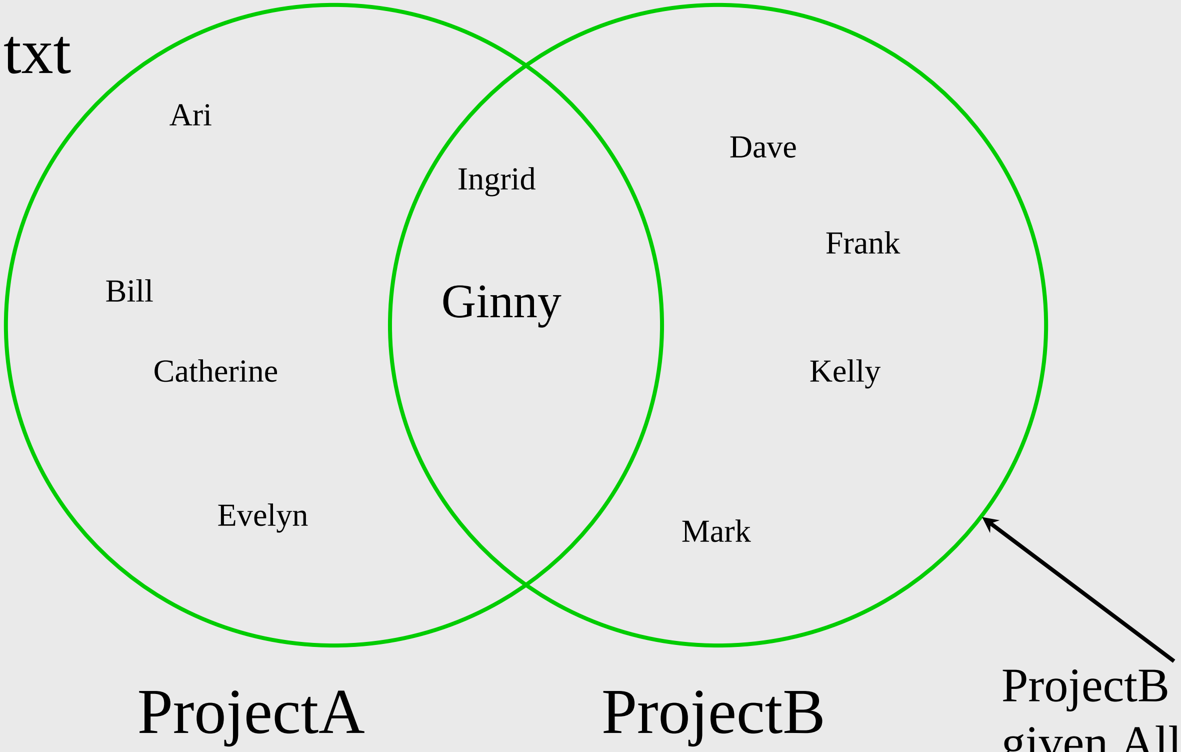
Permissions for file projectAdata.txt



- 🟩 Has Allow Read permission
- 🟥 Has Deny Read permission
- ⬛ Has no Read permission

Ginny task – correct goal state A

Permissions for file projectAdata.txt



ProjectB has been given Allow-Read permission

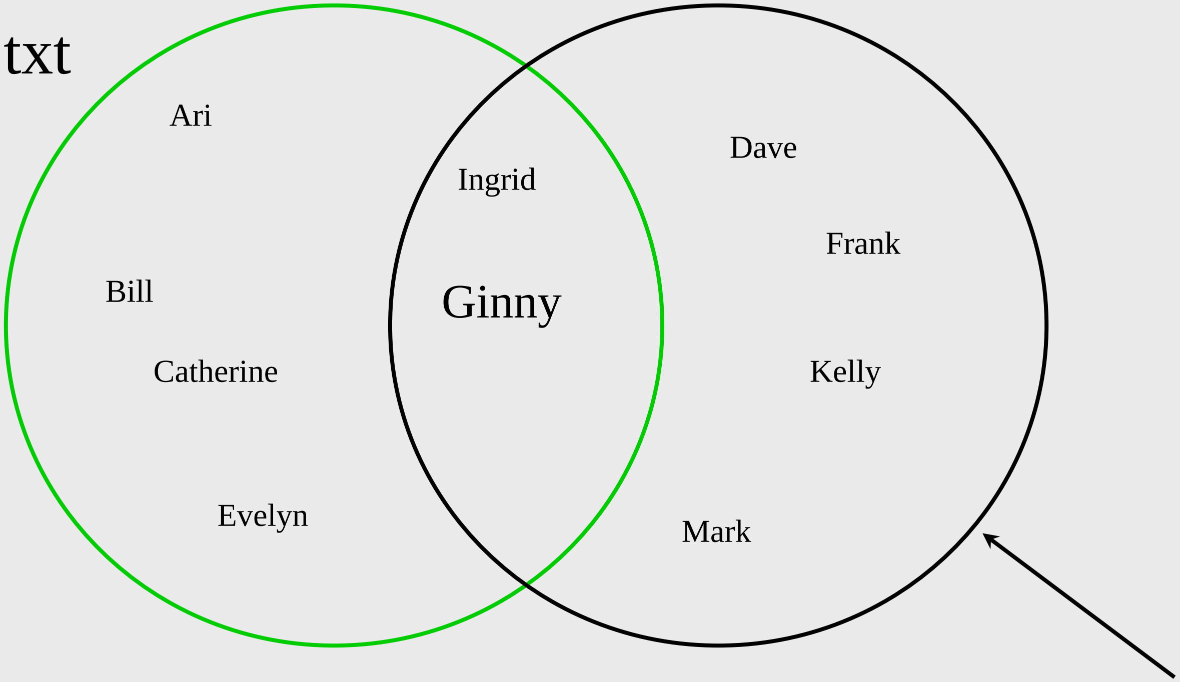
🟩 Has Allow Read permission

🟥 Has Deny Read permission

⬛ Has no Read permission

Ginny task – correct goal state B

Permissions for file projectAdata.txt



Ari

Ingrid

Dave

Frank

Bill

Ginny

Catherine

Kelly

Evelyn

Mark

ProjectA

ProjectB

ProjectB has had Deny-Read permission removed

- 🟩 Has Allow Read permission
- 🟥 Has Deny Read permission
- ⬛ Has no Read permission

Ginny task breakdown

- GOAL: Set permission on file so Ginny can read it.
- There are two groups of users, A and B, each containing several users.
- A user can be a member of either or both of these groups.
- One group is allowed permission to read the file, but the other group is denied permission.
- Ginny is a member of both groups. She cannot access the file.
- The job of the administrator is to change the security settings (file permissions) such that Ginny will be allowed to access the file.

Ginny task steps

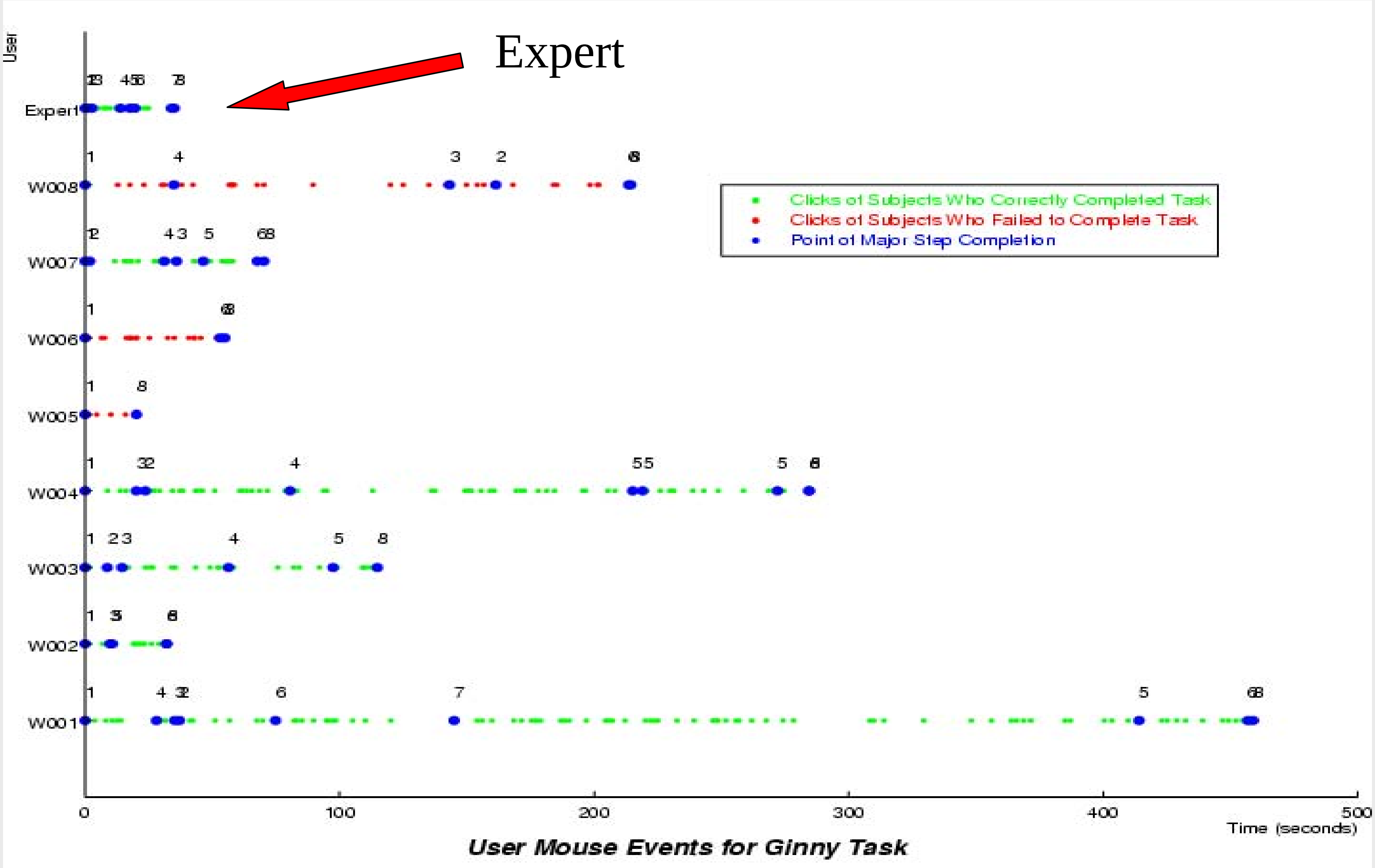
1. Start task
2. Determine ProjectA’s permissions on the data file
3. Determine ProjectB’s permissions on the data file
4. Determine Ginny’s group membership
5. Remove ProjectB’s Deny-Read permission
6. Commit changes
7. Check work
8. End task

Note: Subjects began in the initial position of looking at the properties for the file in question, with the security tab selected … giving them a huge head start.

Data-analysis procedure (user log data)

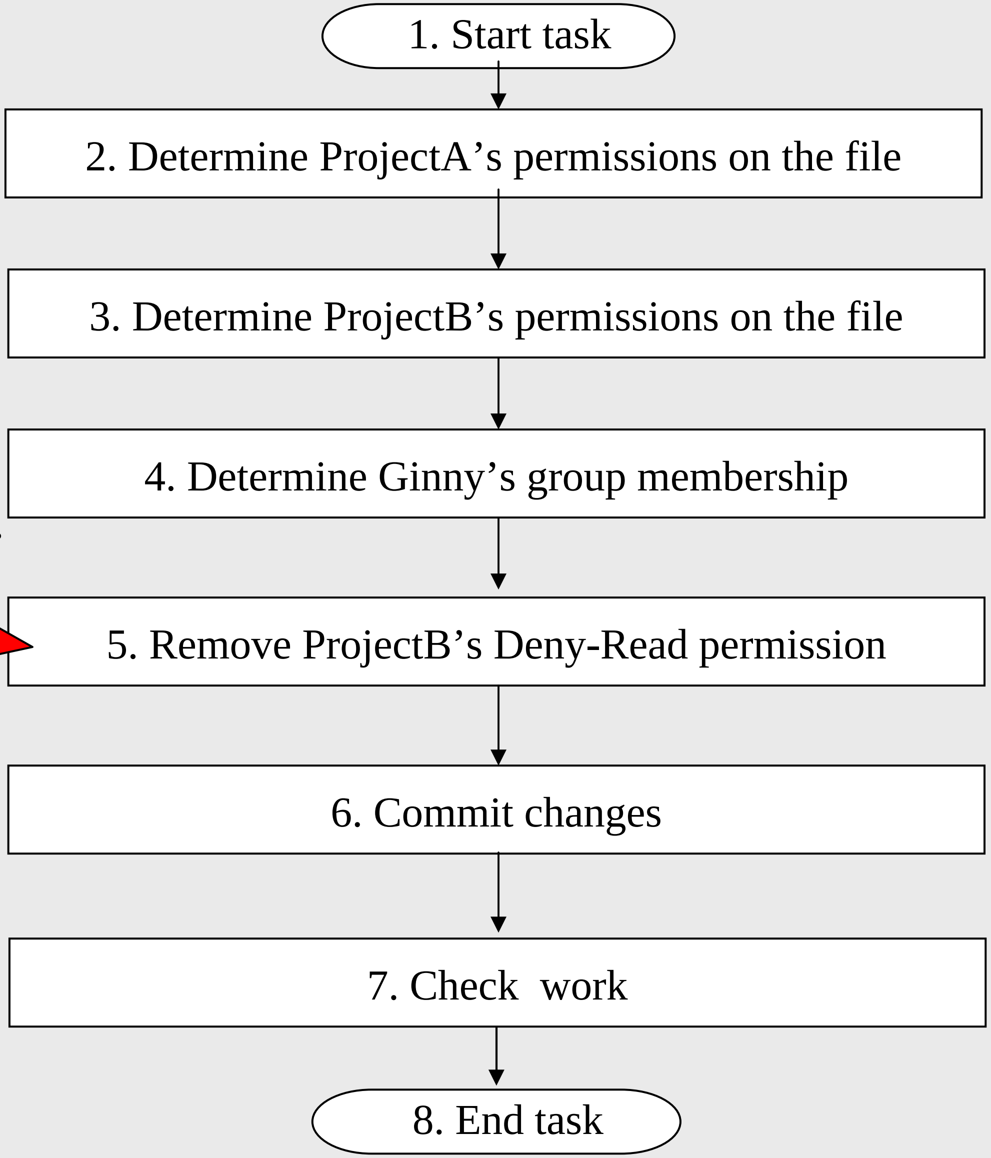
- List steps that expert used to complete task
  - Used verified minimal-step solution
- List mouse clicks required for expert’s steps
- Compare subjects’ clicks with required clicks
- For each click that was not required, categorize it as one of four error categories (after THEA)
- Collapse consecutive clicks of same error type into a single error

Example mouse-click results



Expert

Ginny task steps - accuracy



1. Start task

2. Determine ProjectA’s permissions on the file

3. Determine ProjectB’s permissions on the file

4. Determine Ginny’s group membership

5. Remove ProjectB’s Deny-Read permission

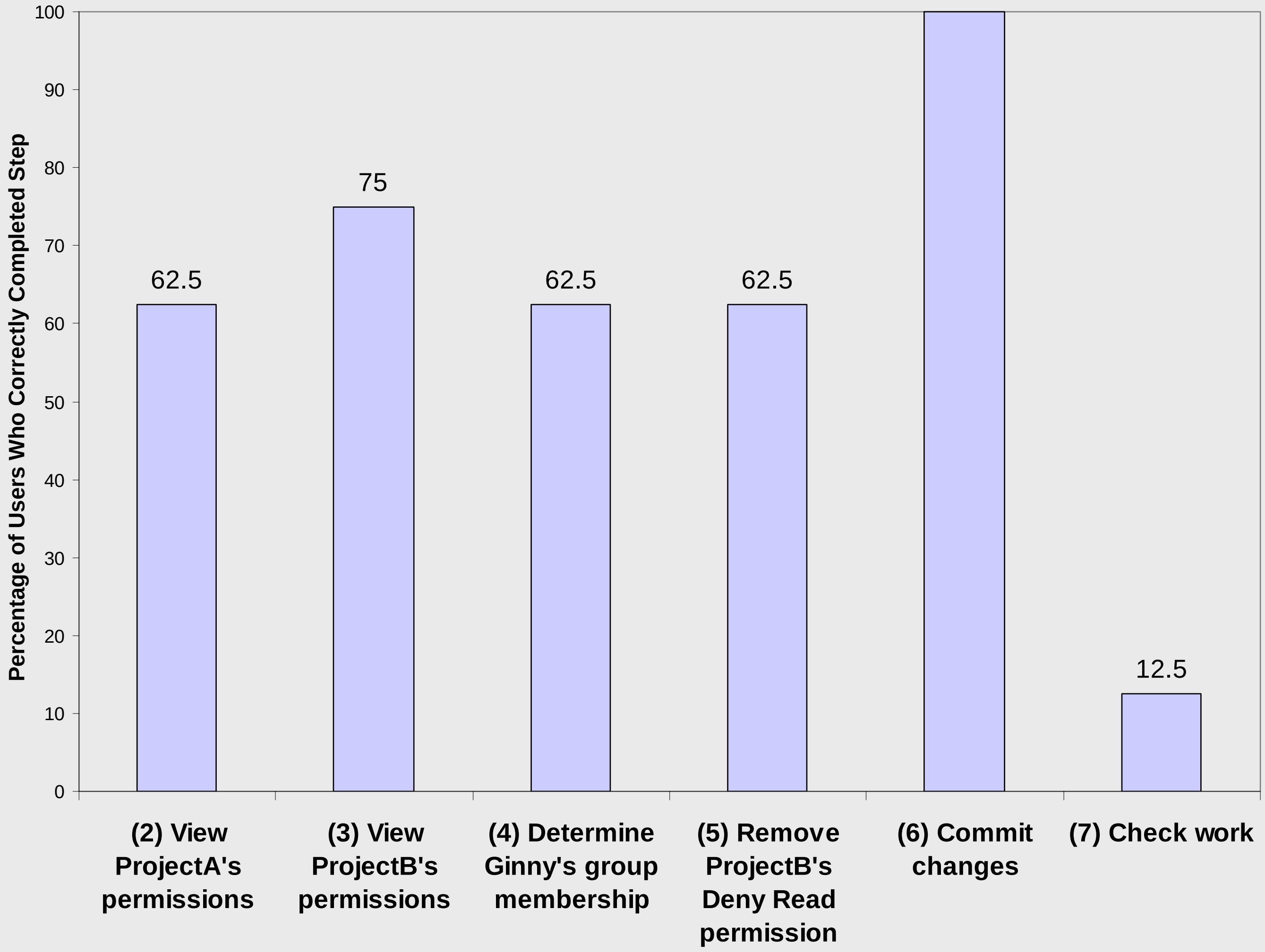
6. Commit changes

7. Check work

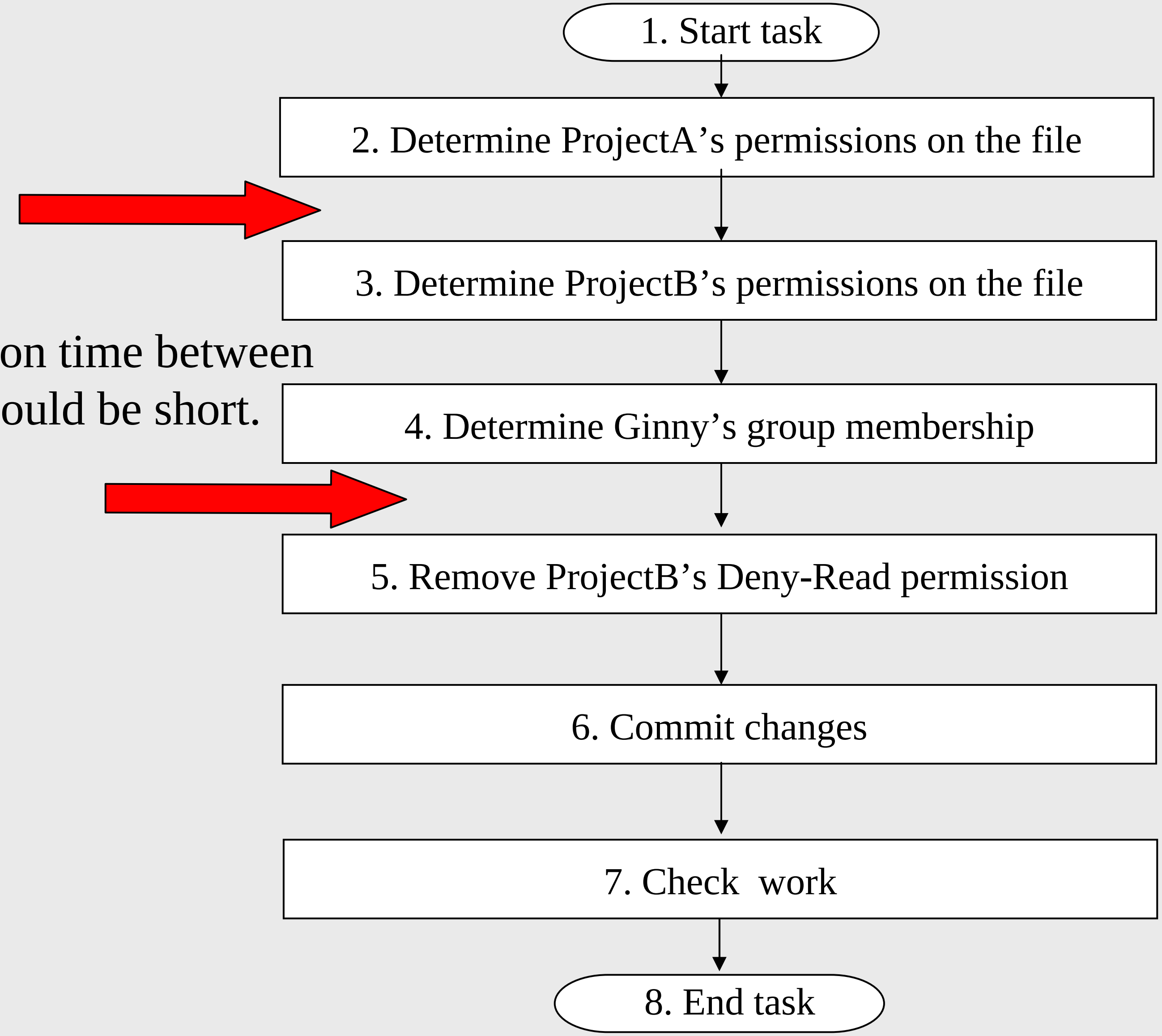
8. End task

Each step should be performed correctly, and in the right order.

Results – accuracy

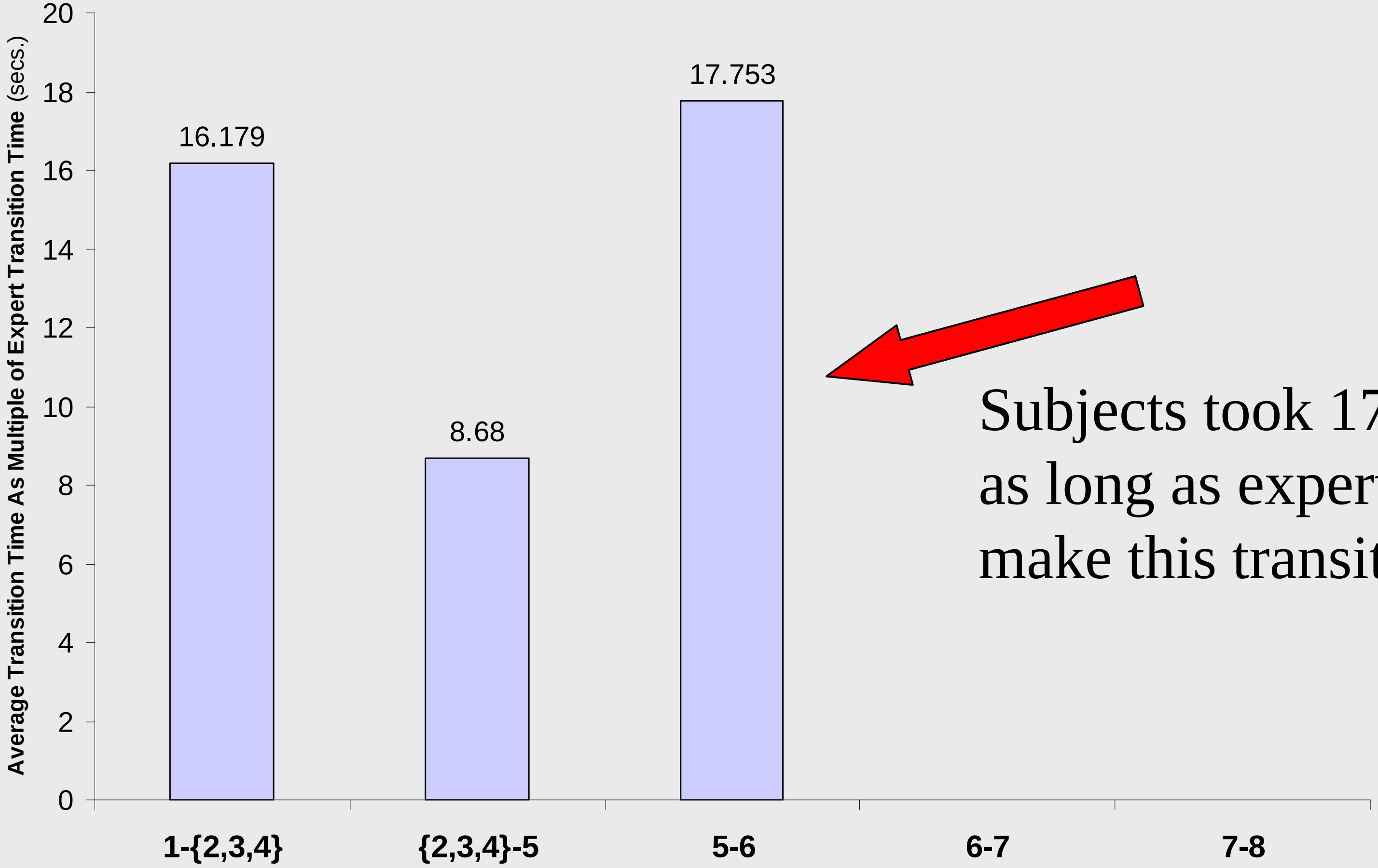


Ginny task steps - speed


1. Start task
2. Determine ProjectA’s permissions on the file
3. Determine ProjectB’s permissions on the file
4. Determine Ginny’s group membership
5. Remove ProjectB’s Deny-Read permission
6. Commit changes
7. Check work
8. End task

Transition time between steps should be short.

Results – speed … Subjects much worse than expert



Subjects took 17 times as long as expert to make this transition.

User transition times as multiples of expert transition times

Explaining these sorry results …

- Users make step-by-step decisions as they progress through the task, enforced to some extent by the user interface.

- Some of these decisions may have been made in error, causing degradations in speed and accuracy.

- What form did those errors take … and what caused them?

Error taxonomy (preliminary; derived from THEA)

- Goal error: An inability to decide what to do, or a decision to do something not on the path to correct completion.
- Plan error: The choice of an action sequence that does not accomplish the intended goal or is impossible to implement.
- Action error: Incorrect implementation of an action sequence. Includes skill-based errors such as slips and lapses.
- Perception error: Failure to perceive needed information (as displayed on screen), incorrect perception of needed information, misinterpretation of information perceived.

Classifying user behaviors as errors

- User behavior was classified as either an error event or a benign event.
  - Error event – a click or a click series that deviated from an experimenter-determined “correct” minimal-click path.
  - Benign event – Correct and “housekeeping” clicks.

Classifying errors into types

- Each error type has its own classification rules; e.g., we classify user behavior as a “goal error” if:
  - User's goal is wrong
    - inferred from user statement (e.g., “I’m changing permissions on user Ginny” … as opposed to permissions on a file)
    - inferred from user actions (e.g., click add; select Ginny)
  - User has no goal
    - inferred from observation of user reading help file to figure out what to do, repeatedly re-reading task statement, or clicking around randomly

Classification rules (plan, action, perception)

- Classify user behavior as a <u>plan error</u> if:
  - User's goal is appropriate, but steps to carry it out may deviate from correct steps.
- Classify as an <u>action error</u> if:
  - User appeared to be carrying out a plan, but deviated from that plan on a single step (e.g., slip or lapse).
  - Note: Even if the plan was wrong, a deviation from that plan would have been categorized as an action error.
- Classify as a <u>perception</u> error if:
  - User fails to notice a needed piece of information, even though it appears on screen.
  - User misinterprets or doesn’t understand a label.

Tally of errors committed

	Error Types			
Subject ID	Goal	Plan	Action	Perception
W001	5	3	2	1
W002	2	0	0	0
W003	4	2	1	0
W004	8	2	0	1
W005	1	0	0	0
W006	1	0	1	0
W007	2	2	1	2
W008	1	1	0	0
Total	24	10	5	4

Count of each of the 4 types of error made by subjects in the Ginny task

Goal and plan errors dominated

- All task failures were caused by a wrong goal or a wrong plan. (Influences accuracy.)
- The remainder of subjects exceeded reasonable expectations in terms of time spent in formulating goals or in pursuing wrong plans. (Influences speed.)

Causes of errors (preliminary)

- Goal error:
  - Inaccessible declarative or procedural knowledge
  - Lack of adequate cues (objects or events suggestive of paths, actions or facts forgotten or imperfectly learned – Note: recognition is easier than recall)
- Plan error:
  - Inaccessible declarative or procedural knowledge
  - Lack of adequate cues (objects or events suggestive of paths, actions or facts forgotten or imperfectly learned – Note: recognition is easier than recall)
- Action error:
  - Imprecision in sensorimotor function (eye/hand)
    - Check boxes in too close proximity
- Perception error:
  - Misperception or misinterpretation of objects or events
    - Definitions of (possibly ambiguous) terms misunderstood or confused

Declarative knowledge (facts)

- To be successful, users must know about …
  - Inheritance – users inherit privileges from groups
  - Precedence – resolution rules for conflicting privileges
  - Effective permissions – the privileges you set may not be the privileges you get

Inheritance

- Users *inherit* permissions from the groups of which they are members.

- Thus, a user may have no explicit Read permission, but if the user is a member of a group with Allow-Read permission on a file, the user will also have Allow-Read permission on that same file.

Precedence rules

- What if a user has permission conflicts?
  - Simultaneous Allow- and Deny-permission (e.g., deriving from membership in multiple groups)
- A *precedence* rule resolves such conflicts.
- Several different precedence rules would appear reasonable to a user (but the correct rule is not made explicit in XP).

Candidate precedence rules

1. Deny takes precedence

   ~~Allow Read from ProjectA~~

   Deny Read from ProjectB

   + ~~Allow Read from Ginny~~

   Deny Read

2. User takes precedence

   ~~Allow Read from ProjectA~~

   ~~Deny Read from ProjectB~~

   + Allow Read from Ginny

   Allow Read

3. Majority-rule precedence

   Allow Read from ProjectA

   ~~Deny Read from ProjectB~~

   + Allow Read from Ginny

   Allow Read

4. List-order precedence

   Allow Read from ProjectA

   ~~Deny Read from ProjectB~~

   + ~~Allow Read from Ginny~~

   Allow Read

Effective vs. stated permissions

- The distinction follows directly from inheritance:
  - Stated permissions
    - The permissions a user gets directly from his/her entry on the permissions list
  - Effective permissions
    - The permissions a user gets by “combining” his/her inherited permissions to his/her stated permissions … in accordance with precedence rules.

- In the absence of inheritance, a user’s effective permissions and stated permissions would be equivalent.

- Effective permissions reflect the actual access a user will have to a file.

Procedural knowledge (how-to-do-it)

- To be successful, users must know about …
  - Procedures for checking results
    - How to check existing permissions on files
    - How to check group memberships
    - How to remove Deny-Read permission
    - How to check effective permissions

Checking outcomes (effective permissions)

- Many users, unsure of themselves, tried to check their work to ensure that the permissions had been set correctly.

- They were largely unable to do so, because the XP mechanism for checking effective permissions is hidden and cumbersome.

  (Only one subject was successful in checking.)

Overcoming the errors

- How could these errors have been avoided?
  - Goal errors
    - provide relevant declarative knowledge … &/or cues
  - Plan errors
    - provide relevant procedural knowledge … &/or cues
  - Action errors
    - attention to colors, juxtapositions, manual devices, etc.
  - Perception errors
    - use of commonly-understood terms and definitions
    - use of symbols and shapes with clear semantics

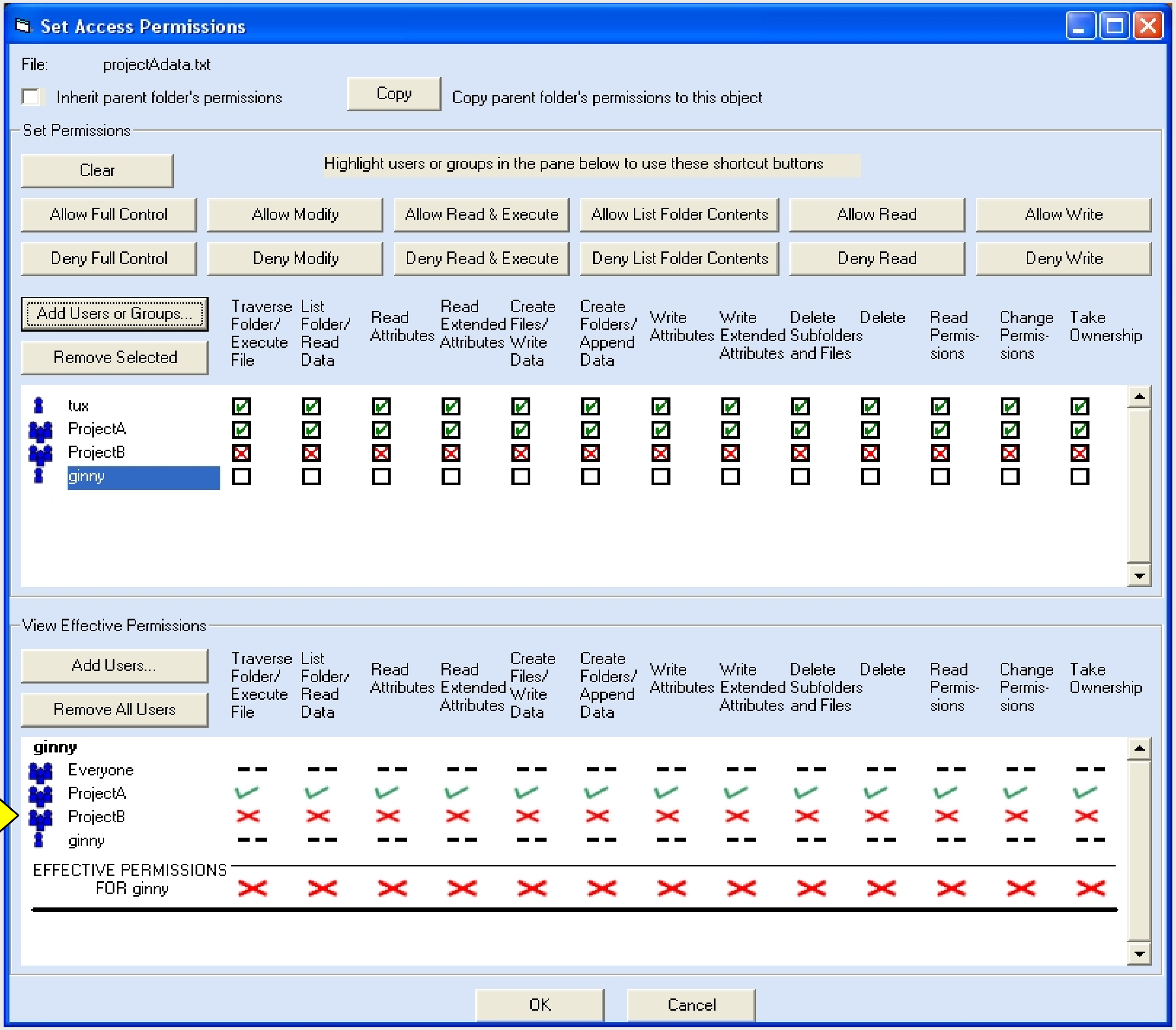
… all with a bias toward avoiding error

Improved "permission-setting" interface (i)

- Declarative knowledge … cues
  - Inheritance - Lower pane (view effective permissions) suggests that Ginny is inheriting permissions … by displaying all the sources of her permissions in one place.
  - Precedence - The view-effective-permissions pane allows for quick trial-and-error testing of candidate precedence rules.
  - Effective permissions - Upper pane (set permissions) juxtaposed with lower pane (view effective permissions) suggests that there is a difference between stated and effective permissions.

Salmon – (inheritance cues)

Ginny’s group membership is on the same screen where permissions are set.



File: projectAdata.txt

□ Inherit parent folder's permissions

Copy parent folder's permissions to this object

Set Permissions

Highlight users or groups in the pane below to use these shortcut buttons

	Traverse Folder/ Execute File	List Folder/ Read Data	Read Attributes	Read Extended Attributes	Create Files/ Write Data	Create Folders/ Append Data	Write Attributes	Write Extended Attributes	Delete Subfolders and Files	Delete	Read Permissions	Change Permissions	Take Ownership
tux	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑
ProjectA	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑
ProjectB	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒
ginny	□	□	□	□	□	□	□	□	□	□	□	□	□

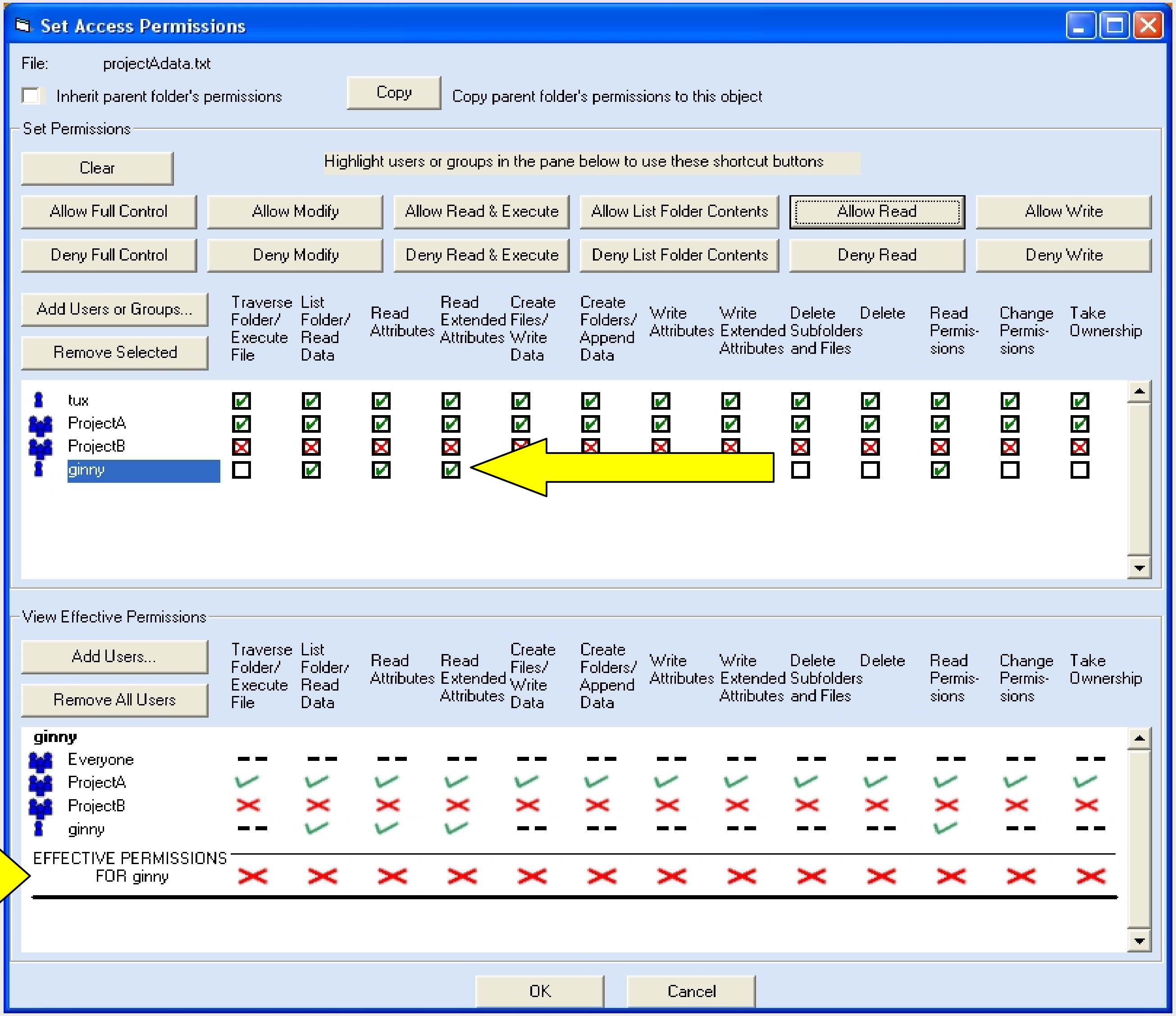
View Effective Permissions

	Traverse Folder/ Execute File	List Folder/ Read Data	Read Attributes	Read Extended Attributes	Create Files/ Write Data	Create Folders/ Append Data	Write Attributes	Write Extended Attributes	Delete Subfolders and Files	Delete	Read Permissions	Change Permissions	Take Ownership
ginny													
Everyone	--	--	--	--	--	--	--	--	--	--	--	--	--
ProjectA	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ProjectB	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
ginny	--	--	--	--	--	--	--	--	--	--	--	--	--
EFFECTIVE PERMISSIONS FOR ginny	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗

Salmon – (precedence cues)

A user who assumes the User Takes Precedence rule will set Ginny’s permissions as shown by the upper arrow.

But a look at Ginny’s effective permissions will show that she still can’t read the file.



Set Access Permissions

File: projectAdata.txt

□ Inherit parent folder's permissions

Copy parent folder's permissions to this object

Set Permissions

Highlight users or groups in the pane below to use these shortcut buttons

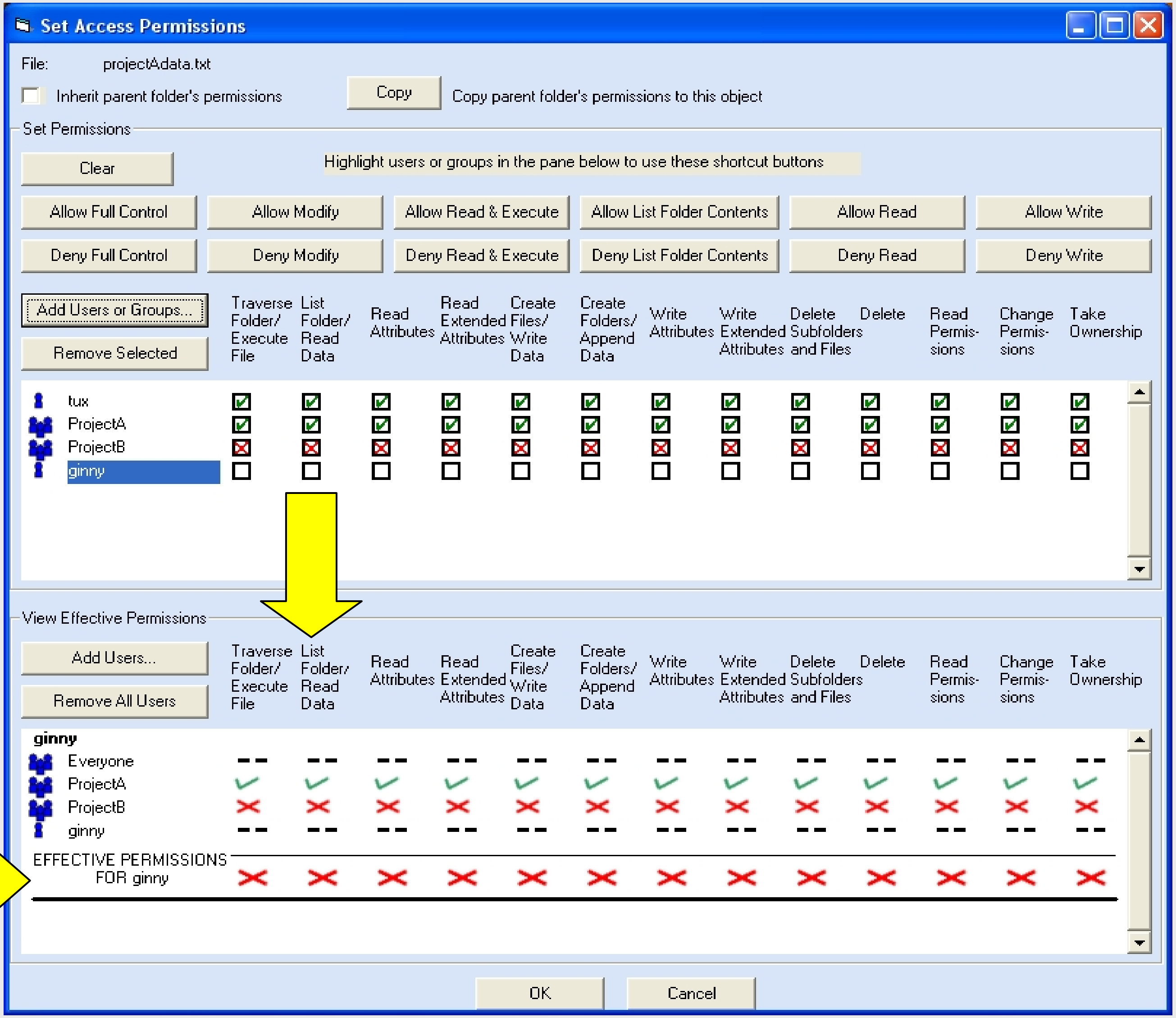
	Traverse Folder/ Execute File	List Folder/ Read Data	Read Attributes	Read Extended Attributes	Create Files/ Write Data	Create Folders/ Append Data	Write Attributes	Write Extended Attributes	Delete Subfolders and Files	Delete	Read Permis- sions	Change Permis- sions	Take Ownership
tux	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑
ProjectA	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑
ProjectB	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒
ginny	□	☑	☑	☑					□	□	☑	□	□

View Effective Permissions

ginny	Traverse Folder/ Execute File	List Folder/ Read Data	Read Attributes	Read Extended Attributes	Create Files/ Write Data	Create Folders/ Append Data	Write Attributes	Write Extended Attributes	Delete Subfolders and Files	Delete	Read Permis- sions	Change Permis- sions	Take Ownership
Everyone	--	--	--	--	--	--	--	--	--	--	--	--	--
ProjectA	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ProjectB	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
ginny	--	✓	✓	✓	--	--	--	--	--	--	✓	--	--
EFFECTIVE PERMISSIONS FOR ginny	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗

Salmon – (effective-permissions cues)

Ginny's effective permissions are visible, and they change as permissions are set in the upper pane.



Set Access Permissions

File: projectAdata.txt

□ Inherit parent folder's permissions

Copy — Copy parent folder's permissions to this object

Set Permissions

Clear

Highlight users or groups in the pane below to use these shortcut buttons

Allow Full Control | Allow Modify | Allow Read & Execute | Allow List Folder Contents | Allow Read | Allow Write

Deny Full Control | Deny Modify | Deny Read & Execute | Deny List Folder Contents | Deny Read | Deny Write

Add Users or Groups... | Remove Selected

	Traverse Folder/ Execute File	List Folder/ Read Data	Read Attributes	Read Extended Attributes	Create Files/ Write Data	Create Folders/ Append Data	Write Attributes	Write Extended Attributes	Delete Subfolders and Files	Delete	Read Permissions	Change Permissions	Take Ownership
tux	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑
ProjectA	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑
ProjectB	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒
ginny	□	□	□	□	□	□	□	□	□	□	□	□	□

View Effective Permissions

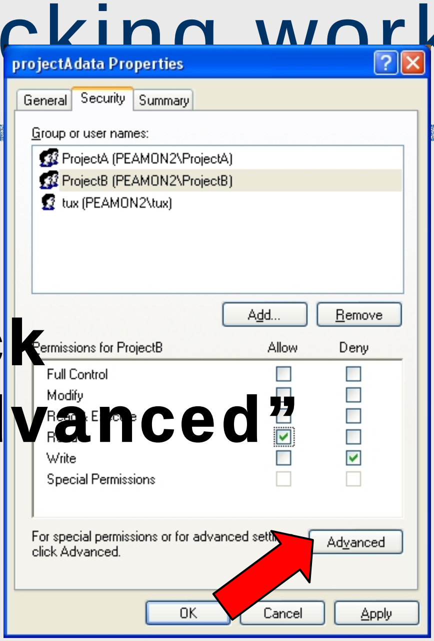
Add Users... | Remove All Users

ginny	Traverse Folder/ Execute File	List Folder/ Read Data	Read Attributes	Read Extended Attributes	Create Files/ Write Data	Create Folders/ Append Data	Write Attributes	Write Extended Attributes	Delete Subfolders and Files	Delete	Read Permissions	Change Permissions	Take Ownership
Everyone	--	--	--	--	--	--	--	--	--	--	--	--	--
ProjectA	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ProjectB	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
ginny	--	--	--	--	--	--	--	--	--	--	--	--	--
EFFECTIVE PERMISSIONS FOR ginny	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗

OK | Cancel

Checking work in the XP interface

1



projectAdata Properties

General | Security | Summary

Group or user names:

- ProjectA (PEAMON2\ProjectA)
- ProjectB (PEAMON2\ProjectB)
- tux (PEAMON2\tux)

Add... Remove

Permissions for ProjectB	Allow	Deny
Full Control	□	□
Modify	□	□
Read & Execute	□	□
Read	☑	□
Write	□	☑
Special Permissions	□	□

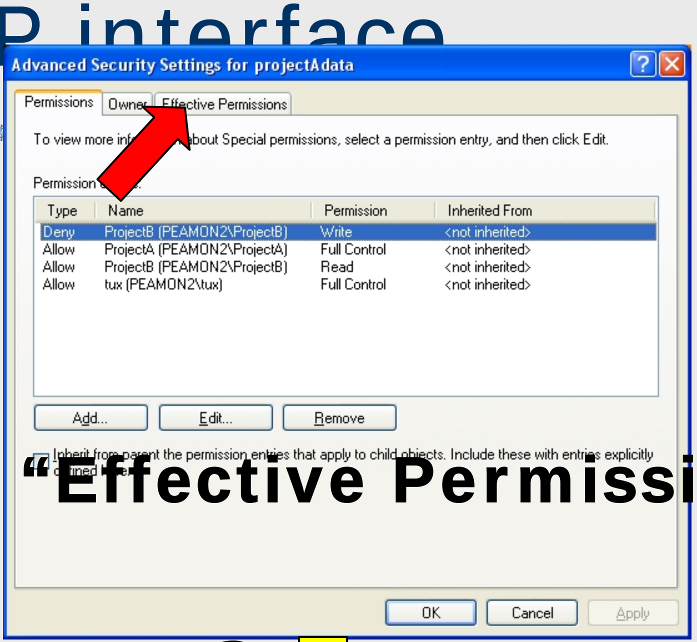
For special permissions or for advanced settings, click Advanced.

Advanced

OK Cancel Apply

Click “Advanced”

2



Advanced Security Settings for projectAdata

Permissions | Owner | Effective Permissions

To view more information about Special permissions, select a permission entry, and then click Edit.

Permission entries:

Type	Name	Permission	Inherited From
Deny	ProjectB (PEAMON2\ProjectB)	Write	<not inherited>
Allow	ProjectA (PEAMON2\ProjectA)	Full Control	<not inherited>
Allow	ProjectB (PEAMON2\ProjectB)	Read	<not inherited>
Allow	tux (PEAMON2\tux)	Full Control	<not inherited>

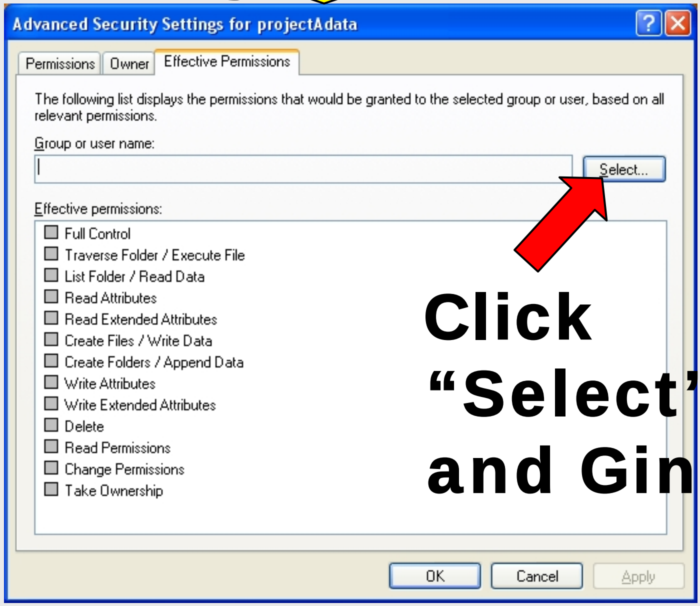
Add... Edit... Remove

□ Inherit from parent the permission entries that apply to child objects. Include these with entries explicitly defined here.

OK Cancel Apply

Click “Effective Permissions”

3



Advanced Security Settings for projectAdata

Permissions | Owner | Effective Permissions

The following list displays the permissions that would be granted to the selected group or user, based on all relevant permissions.

Group or user name:

Select...

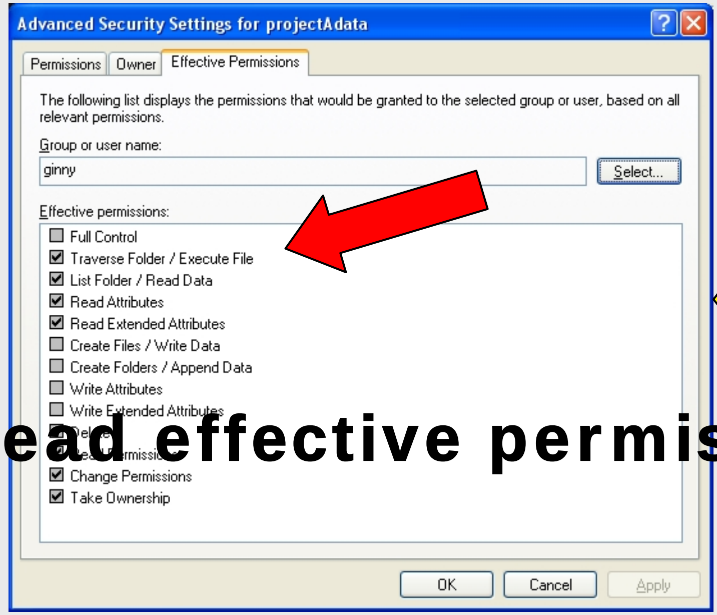
Effective permissions:

- □ Full Control
- □ Traverse Folder / Execute File
- □ List Folder / Read Data
- □ Read Attributes
- □ Read Extended Attributes
- □ Create Files / Write Data
- □ Create Folders / Append Data
- □ Write Attributes
- □ Write Extended Attributes
- □ Delete
- □ Read Permissions
- □ Change Permissions
- □ Take Ownership

OK Cancel Apply

Click “Select” and Ginny

4



Advanced Security Settings for projectAdata

Permissions | Owner | Effective Permissions

The following list displays the permissions that would be granted to the selected group or user, based on all relevant permissions.

Group or user name:

ginny

Select...

Effective permissions:

- □ Full Control
- ☑ Traverse Folder / Execute File
- ☑ List Folder / Read Data
- ☑ Read Attributes
- ☑ Read Extended Attributes
- □ Create Files / Write Data
- □ Create Folders / Append Data
- □ Write Attributes
- □ Write Extended Attributes
- □ Delete
- ☑ Read Permissions
- ☑ Change Permissions
- ☑ Take Ownership

OK Cancel Apply

Read effective permissions

Improved “permission-setting” interface (ii)

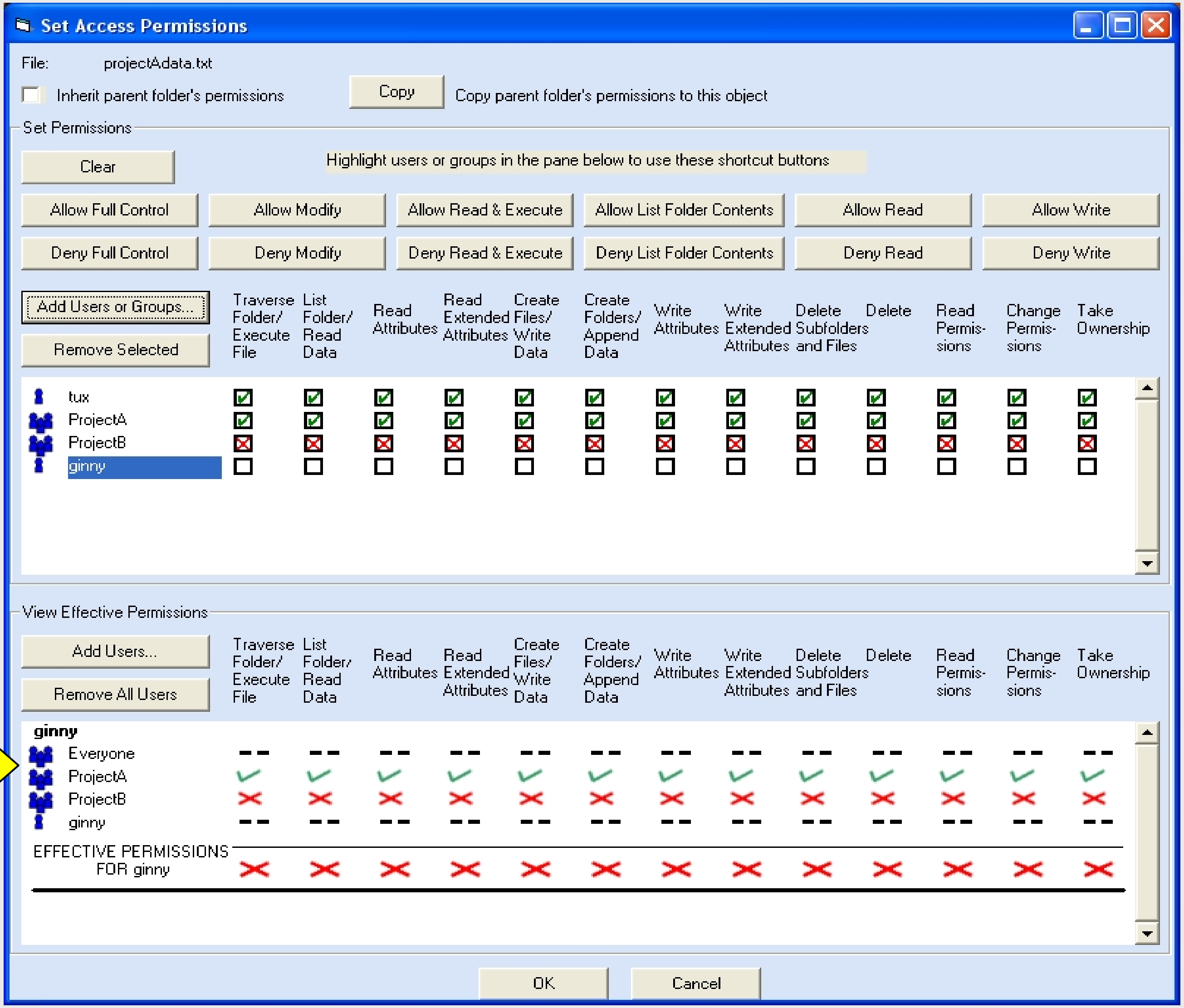
- Procedural knowledge … cues
  - How to begin the process
    - Not clear in XP
  - How to check existing permissions on files
    - All permissions are visible simultaneously (in XP, need to click on each one individually)
  - How to check group memberships
    - Group memberships are shown explicitly
  - How to remove Deny-Read permission
    - A distinct checkbox appears for this
  - How to check effective permissions
    - The view-effective-permissions pane allows for immediate checking of outcome; no need to hunt.

Improved “permission-setting” interface (iii)

- Other cues …
  - Temporal cue for exception, error or status checking
    - When Ginny user name is added in upper pane, Ginny immediately appears in lower pane, drawing the user’s attention toward effective permissions; Ginny appears in both set-permissions pane and effective-permissions pane, prompting the user to look in the lower pane to examine what Ginny’s effective permissions are and what inherited permissions she has from elsewhere.
  - Temporal cue for progress checking
    - When changes are made in the upper pane, they are reflected in the lower pane; perceived movement draws attention to errors, if any, by looking at the effective permissions. Trial-and-error work is reasonably efficient.
  - Visual cue for what to do next
    - Add-Users-or-Groups button is prominent and salient.
    - Lower pane has all the information –- in one place -- that the user needed to collect in XP steps 2-3-4.

Salmon – (temporal cue for error/status checking)

After the user adds Ginny to the upper pane, Ginny also appears in the lower pane, thus directing the user’s attention to the group memberships and effective permissions.



Set Access Permissions

File: projectAdata.txt

□ Inherit parent folder's permissions [Copy] Copy parent folder's permissions to this object

Set Permissions

[Clear] Highlight users or groups in the pane below to use these shortcut buttons

[Allow Full Control] [Allow Modify] [Allow Read & Execute] [Allow List Folder Contents] [Allow Read] [Allow Write]

[Deny Full Control] [Deny Modify] [Deny Read & Execute] [Deny List Folder Contents] [Deny Read] [Deny Write]

[Add Users or Groups...] [Remove Selected]

	Traverse Folder/ Execute File	List Folder/ Read Data	Read Attributes	Read Extended Attributes	Create Files/ Write Data	Create Folders/ Append Data	Write Attributes	Write Extended Attributes	Delete Subfolders and Files	Delete	Read Permis- sions	Change Permis- sions	Take Ownership
tux	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑
ProjectA	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑
ProjectB	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒
ginny	□	□	□	□	□	□	□	□	□	□	□	□	□

View Effective Permissions

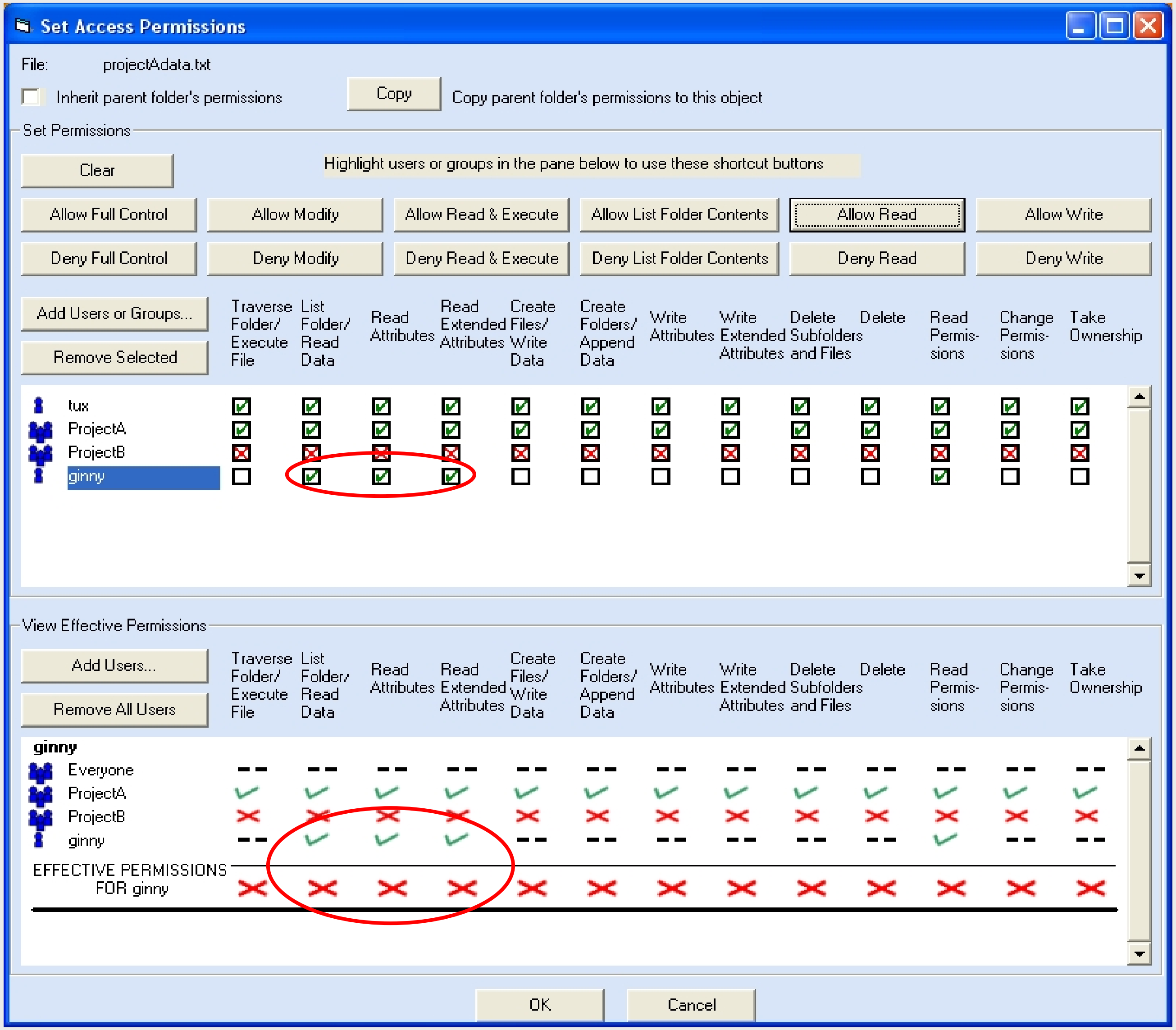
[Add Users...] [Remove All Users]

ginny	Traverse Folder/ Execute File	List Folder/ Read Data	Read Attributes	Read Extended Attributes	Create Files/ Write Data	Create Folders/ Append Data	Write Attributes	Write Extended Attributes	Delete Subfolders and Files	Delete	Read Permis- sions	Change Permis- sions	Take Ownership
Everyone	--	--	--	--	--	--	--	--	--	--	--	--	--
ProjectA	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ProjectB	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
ginny	--	--	--	--	--	--	--	--	--	--	--	--	--
EFFECTIVE PERMISSIONS FOR ginny	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗

[OK] [Cancel]

Salmon – (temporal cue for progress checking)

Changes made in the upper pane are immediately reflected in the lower pane.



Set Access Permissions

File: projectAdata.txt

□ Inherit parent folder's permissions Copy Copy parent folder's permissions to this object

Set Permissions

Clear Highlight users or groups in the pane below to use these shortcut buttons

Allow Full Control | Allow Modify | Allow Read & Execute | Allow List Folder Contents | Allow Read | Allow Write

Deny Full Control | Deny Modify | Deny Read & Execute | Deny List Folder Contents | Deny Read | Deny Write

Add Users or Groups... Remove Selected

	Traverse Folder/ Execute File	List Folder/ Read Data	Read Attributes	Read Extended Attributes	Create Files/ Write Data	Create Folders/ Append Data	Write Attributes	Write Extended Attributes	Delete Subfolders and Files	Delete	Read Permissions	Change Permissions	Take Ownership
tux	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑
ProjectA	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑
ProjectB	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒
ginny	□	☑	☑	☑	□	□	□	□	□	□	☑	□	□

View Effective Permissions

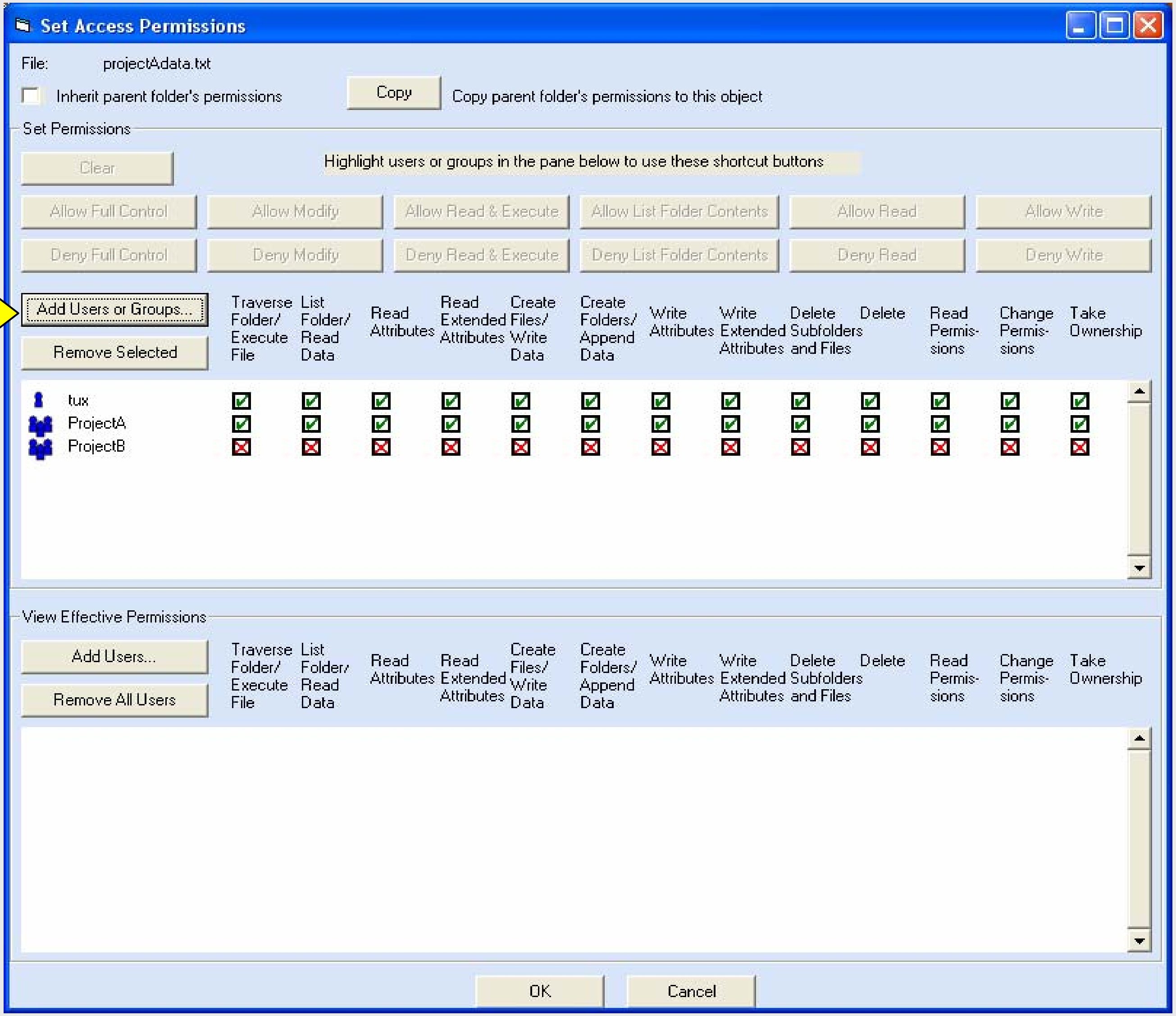
Add Users... Remove All Users

	Traverse Folder/ Execute File	List Folder/ Read Data	Read Attributes	Read Extended Attributes	Create Files/ Write Data	Create Folders/ Append Data	Write Attributes	Write Extended Attributes	Delete Subfolders and Files	Delete	Read Permissions	Change Permissions	Take Ownership
ginny													
Everyone	--	--	--	--	--	--	--	--	--	--	--	--	--
ProjectA	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ProjectB	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
ginny	--	✓	✓	✓	--	--	--	--	--	--	✓	--	--
EFFECTIVE PERMISSIONS FOR ginny	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗

OK Cancel

Salmon – (visual cue for what to do next - i)

The starting state of the Salmon interface for the Ginny task. Note the “Add Users or Groups” button – a prominent cue as to where to start.



File: projectAdata.txt

□ Inherit parent folder's permissions

	Traverse Folder/ Execute File	List Folder/ Read Data	Read Attributes	Read Extended Attributes	Create Files/ Write Data	Create Folders/ Append Data	Write Attributes	Write Extended Attributes	Delete Subfolders and Files	Delete	Read Permis-sions	Change Permis-sions	Take Ownership
tux	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑
ProjectA	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑
ProjectB	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒	☒

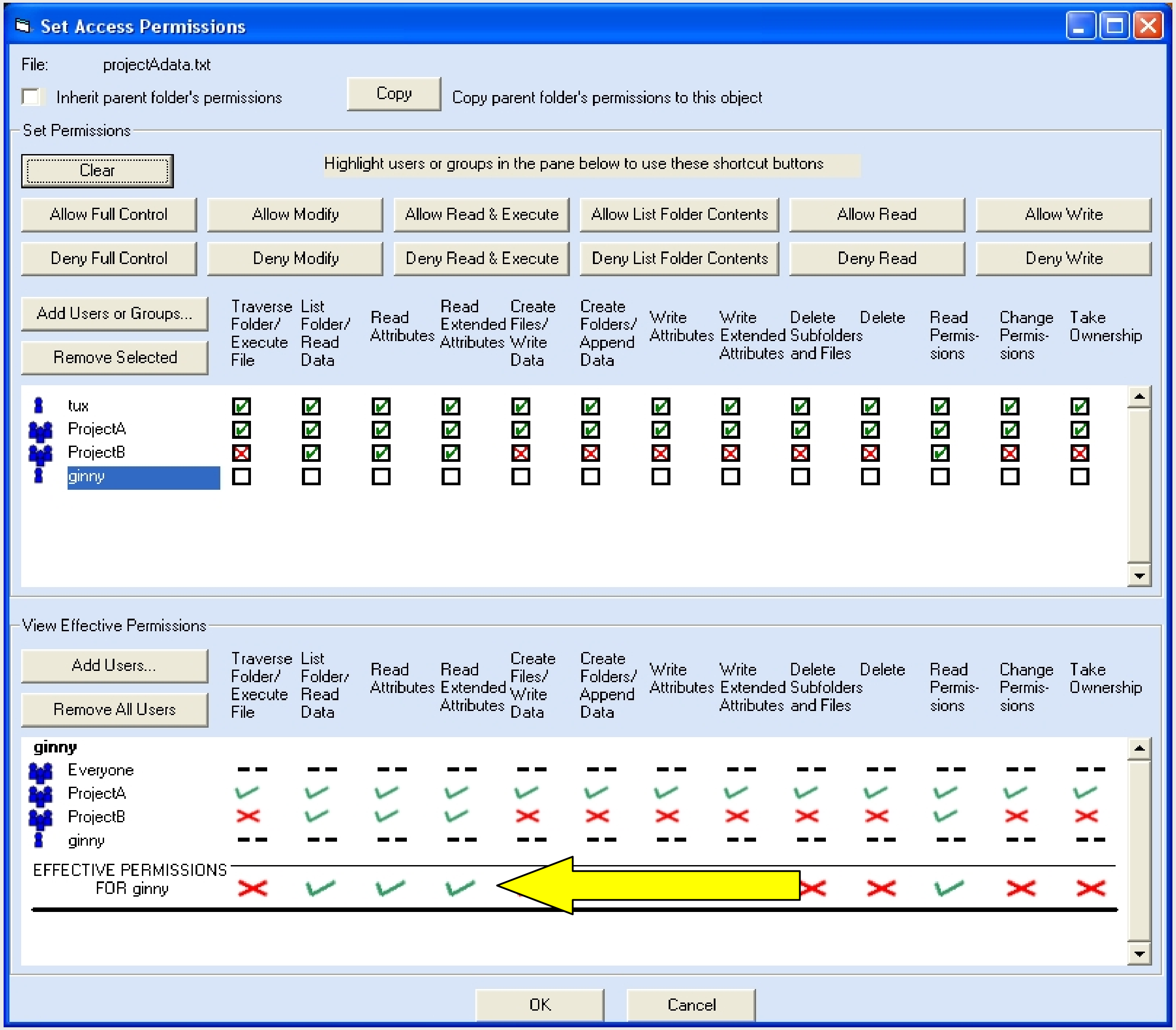
Traverse Folder/ Execute File	List Folder/ Read Data	Read Attributes	Read Extended Attributes	Create Files/ Write Data	Create Folders/ Append Data	Write Attributes	Write Extended Attributes	Delete Subfolders and Files	Delete	Read Permis-sions	Change Permis-sions	Take Ownership

Salmon – (visual cue for what to do next - ii)

Group membership and group permissions are all visible in the lower pane.

Correctness can be verified easily and immediately.

Lower pane has all the necessary info.



Set Access Permissions

File: projectAdata.txt

□ Inherit parent folder's permissions [Copy] Copy parent folder's permissions to this object

Set Permissions

[Clear] Highlight users or groups in the pane below to use these shortcut buttons

[Allow Full Control] [Allow Modify] [Allow Read & Execute] [Allow List Folder Contents] [Allow Read] [Allow Write]

[Deny Full Control] [Deny Modify] [Deny Read & Execute] [Deny List Folder Contents] [Deny Read] [Deny Write]

[Add Users or Groups...] [Remove Selected]

	Traverse Folder/ Execute File	List Folder/ Read Data	Read Attributes	Read Extended Attributes	Create Files/ Write Data	Create Folders/ Append Data	Write Attributes	Write Extended Attributes	Delete Subfolders and Files	Delete	Read Permis-sions	Change Permis-sions	Take Ownership
tux	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑
ProjectA	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑
ProjectB	☒	☑	☑	☑	☒	☒	☒	☒	☒	☒	☑	☒	☒
ginny	□	□	□	□	□	□	□	□	□	□	□	□	□

View Effective Permissions

[Add Users...] [Remove All Users]

ginny	Traverse Folder/ Execute File	List Folder/ Read Data	Read Attributes	Read Extended Attributes	Create Files/ Write Data	Create Folders/ Append Data	Write Attributes	Write Extended Attributes	Delete Subfolders and Files	Delete	Read Permis-sions	Change Permis-sions	Take Ownership
Everyone	--	--	--	--	--	--	--	--	--	--	--	--	--
ProjectA	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ProjectB	✗	✓	✓	✓	✗	✗	✗	✗	✗	✗	✓	✗	✗
ginny	--	--	--	--	--	--	--	--	--	--	--	--	--
EFFECTIVE PERMISSIONS FOR ginny	✗	✓	✓	✓					✗	✗	✓	✗	✗

[OK] [Cancel]

Salmon – preliminary results

- Data still being analyzed
- First results indicate about 30% speed improvement over XP (10x instead of 17 x)
- Accuracy is 100%

Summary

- What was learned?
  - Dependability (speed / accuracy) is affected by goals, plans, actions and perceptions.
  - Declarative knowledge is needed
  - Procedural knowledge is needed
  - Availability and salience of cues are important
- Sources of error (undependability) were identified.
- We could design dependability into an interface if we attend to these notions at design time, perhaps through the use of a modified HTA, or THEA or other evaluation tool.
- Cognitive biases and human error are important factors in dependable HCI.

Conclusion

- There is an explanation for what users do.
- Users do "the wrong thing" for reasons, not usually random.
- We can ascertain these reasons
- We can, knowing the reasons, design user interfaces in ways that influence user behavior so that the probability of users doing "the right thing" is higher and more frequent than of their doing "the wrong thing."

References

- Norman, Donald A. *The Psychology of Everyday Things*. Basic Books, 1988.
- Pocock, Steven; Fields, Bob; Harrison, Michael; and Wright, Peter. *THEA - A Reference Guide*. University of York Technical Report YCS-2001-336, 2001.
- Reason, James. *Human Error*. Cambridge University Press, 1990.
- U.S. Senate Sergeant at Arms. *Report on the Investigation into Improper Access to the Senate Judiciary Committee's Computer System*. March, 2004.
  http://judiciary.senate.gov/testimony.cfm?id=1085&wit_id=2514